UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

FACULTAT D'INFORMÀTICA DE BARCELONA
GRAU EN ENGINYERIA INFORMÀTICA
ESPECIALITAT COMPUTACIÓ

ARTICLE SEARCH ENGINE

TREBALL DE FI DE GRAU
MEMÒRIA FINAL

Autor: **Aguilar Barroso, Manuel**
Director: **Carbonell Batlle, Ernest** (Tiendeo)
Codirector: **Pereira Vieito, Pedro José** (Tiendeo)
Ponent: **Barlet Ros, Pere** (AC)

7 de gener, 2022

Resum
Aquest treball de fi de grau és un projecte per desenvolupar i canviar el cercador de la pàgina web de Tiendeo.com. El cercador del qual l'empresa Tiendeo disposa, és un que permet a l'usuari cercar per paraules precalculades i no cercar lliurement. A mesura que Tiendeo se'n va digitalitzant, aquest cercador va perdent qualitat.

Per tant, en aquest projecte, es tracta de crear un nou cercador que millori els problemes que l'antic tenia, canviant l'ordenació de totes les paraules precalculades i donant a l'usuari la possibilitat de poder cercar qualsevol mena text el qual no es trobi precalculat. A més, arrel que l'usuari pot cercar qualsevol mena de text, es dissenyarà un nou sistema d'autocompletat del cercador, amb l'objectiu que l'usuari vegi en tot moment les cerques més buscades.

Llavors, el treball s'emmarca en l'objectiu de millorar un cercador que de mica en mica anava empitjorant, així com donar més opcions a l'usuari i intentar millorar la seva experiència.

Resumen
Este trabajo de fin de grado es un proyecto para desarrollar y cambiar el buscador de la página web de Tiendeo.com. El buscador del que la empresa Tiendeo dispone, es uno que permite al usuario buscar por palabras precalculadas y no buscar libremente. A medida que Tiendeo se va digitalizando, este buscador va perdiendo calidad.

Por lo tanto, en este proyecto, se trata de crear un nuevo buscador que mejore los problemas que el antiguo tenía, cambiando la ordenación de todas las palabras precalculadas y dando al usuario la posibilidad de poder buscar cualquier tipo texto que no se encuentre precalculado. Además, a raíz de que el usuario puede buscar cualquier tipo de texto, se diseñará un nuevo sistema de autocompletado del buscador, con el objetivo de que el usuario vea en todo momento las búsquedas más buscadas.

Entonces, el trabajo se enmarca en el objetivo de mejorar un buscador que paulatinamente iba empeorando, así como dar más opciones al usuario e intentar mejorar su experiencia.

Abstract

This degree final project is a project to develop and change the search engine on the Tiendeo.com website. The search engine that the company Tiendeo has, is one that allows the user to search by precalculated words and not search freely. As Tiendeo is digitizing, this search engine is losing quality.

Therefore, this project is about creating a new search engine that improves the problems that the old one had, changing the results of all the precalculated words and giving the user the possibility of being able to search for any type of text that is not precalculated. In addition, since the user can search for any type of text, a new search engine autocomplete system will be designed, with the aim that the user sees the most searched searches at all times.

So the work is part of the objective of improving a search engine that was gradually getting worse, as well as giving the user more options and trying to improve their experience.

Índex

Introducció i contextualització **10**
Context 11
Conceptes importants 11
Search Engine 11
Entitats a treballar 11
Identificació del problema 13
Actors implicats 13

Justificació del projecte **15**
Estudi de les solucions existents 15
Selecció de programari 16

Abast del projecte **18**
Objectius 18
Requisits 18
Requisits funcionals 18
Requisits no funcionals 19
Obstacles i riscos 19

Metodologia i rigor **21**
Eines de desenvolupament 21
Mètode de validació 21

Planificació temporal **23**
Descripció de les tasques 23
Gestió del projecte 23
Anàlisi de tecnologies 24
Desenvolupament del cercador 24
Anàlisi de resultats 25
Recursos 26
Recursos humans 26
Recursos materials 26
Recursos indirectes 27
Estimació temporal 27
Planificació inicial 28
Planificació final 30
Gestió de risc: Plans alternatius i obstacles 31

Pressupost **33**
Identificació dels costos 33
Recursos humans 33
Recursos materials 35
Recursos indirectes 37
Contingència 39

Estimació dels costos inicial 39
Estimació dels costos final 40
Control de gestió 41

Informe de sostenibilitat 42
Autoavaluació 42
Dimensió ambiental 42
Dimensió econòmica 44
Dimensió social 46

Anàlisi de diferents tecnologies 48
Search engines al mercat 48
Elasticsearch 48
Splunk 49
Solr 50
Search Engine escollit 50
Administradors de contenidors 50

Desenvolupament del cercador 52
Arquitectura del projecte 52
Disseny de les pipelines 52
Distribució de carpetes 54
Visió general del cercador 56
Extracció de les Features 57
Definició 57
Mètode d'extracció 58
Extracció dels Scores 58
Definició 58
Mètode d'extracció 59
Pujada de les dades a ElasticSearch 60
Creació de les Collections 61
Funcionament de l'autocompletat 62
Consumició del contingut 65
Search Engine 65
Autocompletat 65
Jobs automàtics de Kubernetes 66
Selecció de les dades a utilitzar 66
Taules a utilitzar 66
Camps que defineixen un document 68
Definició dels analyzers 70
Configuració per a cada idioma 71
Creació de la query de cerca 73
Boolean Query 74
Must Query 74
Should Query 77
Function Score 78

Query final 78

Validació dels resultats **82**
Qualitat dels resultats 83
Normalized Discounted Cumulative Gain 83
Click Through Rate 85

Integració de coneixements **88**

Lleis i regulacions **89**

Conclusions **90**
Assoliment dels objectius 90
Assoliment competències tècniques 91
Futur del cercador 93
Conclusió final 94

Bibliografia **95**

Índex de figures

Evolució de Google en els darrers vint anys 10
Exemple de Catàleg i articles 12
Opcions limitades de cerca a Tiendeo 13
top 10 Search Engines del mercat 16
Diagrama de Gantt de les tasques a realitzar 29
Diagrama de Gantt de la planificació final de les tasques a realitzar 30
Preu d'Splunk per volum de documents 49
Fases de les pipelines 53
Pipeline per pujar les dades a Elasticsearch 54
Estructura de les carpetes del projecte 55
Procés per calcular les features 58
Procés per calcular els scores 60
Pipeline de la pujada de les dades a Elasticsearch 61
Pipeline per generar les Collections 62
Actual autocompletat del cercador 63
Càlcul de les suggerències i les respectives features 63
Pipeline per calcular la Collection per l'autocompletat 65
Jerarquia dels productes 67
Exemple d'un articles i els seus respectius camps 70
Analyzers per Espanya 72
Exemple d'aplicació dels analyzers 72
Temps d'execució per cerques diferents 76
Exemple de Dashboard d'Streamlit 82
Càlcul DCG d'una llista 83
Càlcul ideal de DCG d'una llista 83
Càlcul nDCG d'una llista 83
Gràfic de barres del nDCG per 20 cerques diferents 84
Distribució del CTR durant el temps 85
Distribució del CTR durant el temps, separant per tipus de cerca 86
Distribució de les navegacions, separant per tipus de cerca 87

Índex de taules

Tasques a realitzar 28
Sou brut per hora per cada rol del projecte 34
Cost dels recursos humans del projecte 35
Amortització de cada recurs material de hardware 36
Amortització del recursos de software 37
Costos de l'espai de treball 38
Amortització de cada recurs indirecte 38
Costos aplicant les contingències 39
Costos dels possibles imprevistos 39
Pressupost del projecte 40
Pes i valor normalitzat de cada feature 59

1. Introducció i contextualització

Els motors de cerca (o *search engines* en anglès) poden ser definits com un sistema de software el qual és encarregat de fer les cerques, generalment a la web[1]. Hi ha molts exemples de cercadors orientats a diferents àmbits, tals com: cercadors per a pàgines web per cercar productes, cercar una aplicació en el telèfon mòbil, cercar pàgines web (Google)… però tots ells tenen una cosa en comú, l'extracció d'informació (*information retrieval*[1]) en un temps ínfim.

En l'actualitat, les cerques a la web han augmentat un 2000% comparat amb els darrers vint anys[3]. Aquest augment, en gran part, es deu al gran progrés tecnològic que hi ha hagut en aquests últims anys. En la figura 1 es pot apreciar l'índex de creixement que ha tingut Google en aquests darrers vint anys de progrés. Es pot observar que és un creixement que de mica en mica s'apropa a ser exponencial.

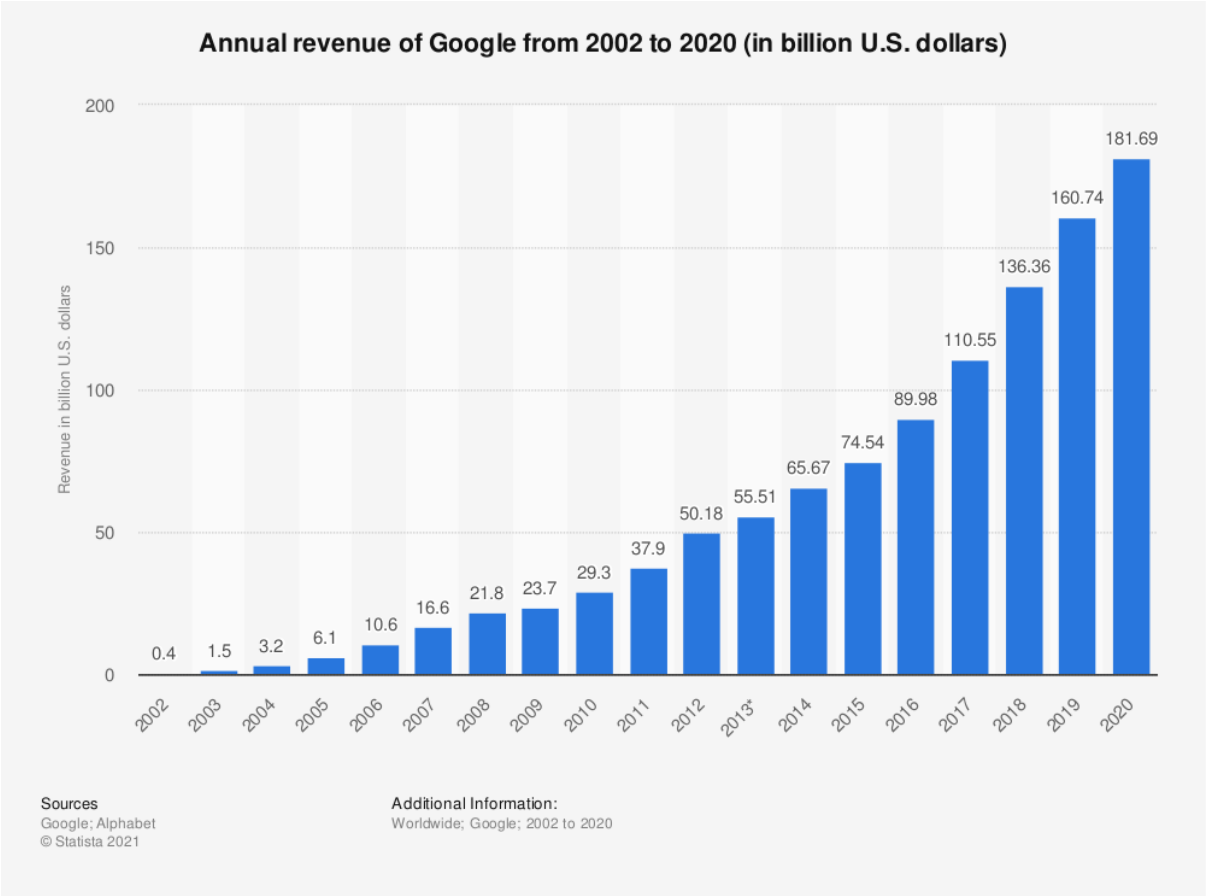


Figura 1: Mostra l'evolució de Google en els darrers vint anys.
Font: statista.com/statistics/266206/googles-annual-global-revenue/

A causa d'aquest augment en cerques a la web, fa que cada vegada hi hagi més competència en el mercat dels *search engines*. Fet que provoca que les cerques

[1] És el procés d'obtenció de recursos del sistema d'informació que són rellevants per a una necessitat d'informació a partir d'una col·lecció d'aquests recursos[2].

siguin més exigents, buscant apropar-se a fer-les pràcticament instantànies i de la qualitat més gran possible.

Per tant, el que es buscarà fer en aquest Treball de Fi de Grau (o TFG, d'aquí en endavant) és fer un cercador, el qual sigui capaç de fer cerques amb la màxima qualitat i rendiment possible.

1.1. Context

En aquest document es detalla de forma precisa com es gestiona aquest TFG. Aquest projecte es troba dins la modalitat B, el que implica que està sent desenvolupat en una empresa, exactament a Tiendeo Web Marketing, S.L. Tiendeo és una empresa que té una web i una app en 44 països diferents i s'encarrega que els usuaris vegin en tot moment tots els tipus d'ofertes sobre diferents productes (d'esport, alimentació, estris…) en les botigues properes a ells.

El projecte consta en desenvolupar un nou cercador per a Tiendeo, és a dir, a partir d'un text d'entrada d'un usuari, oferir una ordenació de productes que s'ofereixen a Tiendeo. Per tant, seria utilitzat principalment pels usuaris els quals vulguin buscar algun producte dins la web o l'app de Tiendeo. L'empresa ja té un cercador el qual et permet buscar paraules precalculades, però es vol fer un de nou amb millor definició que et permeti buscar qualsevol cosa (es veurà amb més profunditat a l'apartat 2. Justificació del projecte).

1.2. Conceptes importants

A continuació es defineixen els conceptes d'especial rellevància per a poder entendre quin enfocament tindrà el cercador a desenvolupar.

1.2.1. Search Engine

Per entrar més en detall del que seria un Search Engine, es defineix com un sistema per extreure la informació que es troba guardada en un ordinador. Aquesta informació extreta usualment és presentada en un format llista i s'anomenen *hits*. Existeixen molts tipus de Search Engines, però el que s'utilitzarà en aquest projecte serà per contingut, és a dir, cerques centrades en el text.

1.2.2. Entitats a treballar

Per a poder entendre amb quines dades treballa el cercador actual i el nou a desenvolupar, es defineixen les següents entitats:

- Catàleg: entitat que conté un o més *articles*, generalment d'un *retailer* o *manufacturer* en específic, per tal que els usuaris de Tiendeo puguin veure de forma agrupada els articles que ofereixen. Es pot veure un exemple de catàleg del retailer FESMÉS en la figura 2.

- Article: És un producte en específic oferit per una *marca, retailer* o *manufacturer*, el qual surt en un catàleg. Un exemple també es podria veure en la mateixa figura 2, on la televisió és un article oferit (ja que té la fletxa blava en la qual es podria clicar i et redirigeix a la pàgina de l'article).

Figura 2: Mostra un exemple de Catàleg i els articles que s'ofereixen.
Font: Tiendeo.com

- Retailer/Manufacturer: Els retailers són empreses les quals venen els productes directament als consumidors (detallistes). Mentre que els manufacturers ho fan a l'engròs (majoristes).

- Marca: Tot article pot tenir una marca. En el cas de la televisió de la figura 2, que és un article i es troba en el catàleg anterior seria Samsung (tot i que no es pot apreciar a la figura).

- Sale: És el mateix que un article, tot i que l'única diferència és que una sale no apareix en cap catàleg.

- Producte: És una forma de descriure un article o sale. Per exemple, un article que sigui televisor LG 40 polzades, el seu producte és "televisor".

1.3. Identificació del problema

En l'actual cercador de Tiendeo no es pot cercar per qualsevol entrada que hi posi l'usuari, ja que et limita sempre a només les paraules que estan precalculades. És a dir, tenen una sèrie extensa de productes, retailers, marques, etc. on, per cada element, està calculada la seva llista a retornar d'articles o sales. Per exemple, fixant-se en la figura 3, si un usuari volgués cercar "tablet 12 polzades samsung" no podria, perquè els únics productes precalculats són els que es veuen en el desplegable (l'autocompletat).

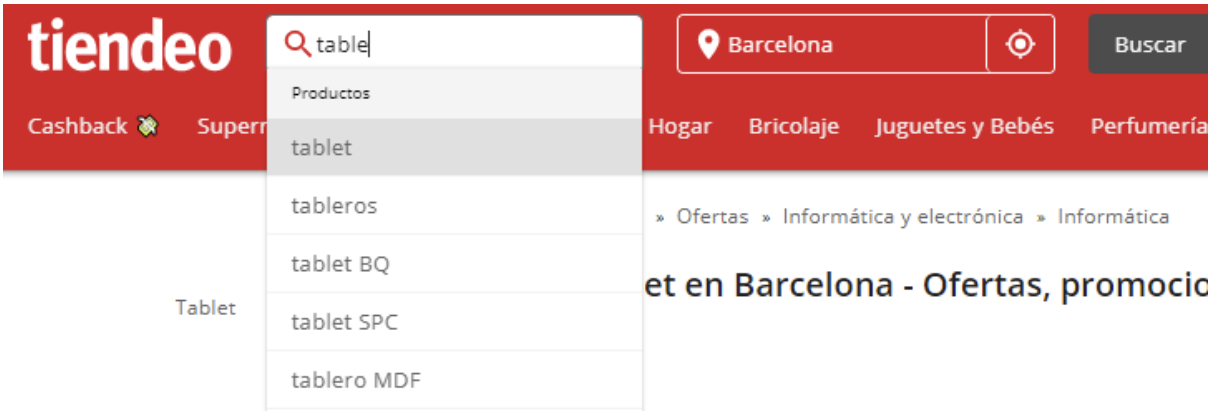


Figura 3: Mostra les opcions limitades que tens per a cercar una tablet.
Font: Tiendeo.com

A més, si un article *televisor LG 40 polzades*, el seu producte estigués mal etiquetat, i en comptes de posar televisor, hi posés tablet, ja no sortiria en la llista de resultats que veurà l'usuari.

Per tant, el problema a resoldre és poder donar a l'usuari la llibertat que pugui cercar qualsevol mena de text i poder ser independents d'errors (siguin generats per humans o per bots) a l'hora de possibles etiquetes mal posades en articles o sales.

1.4. Actors implicats

Aquest projecte té diferents agents implicats i són els següents:

- Usuaris. Són persones les quals entren a Tiendeo per tal de veure les novetats en catàlegs i articles. Seran els principals consumidors del nou cercador.

- Clients. És una de les fonts principals d'ingressos de Tiendeo i són empreses que paguen per promocionar els seus productes. Un exemple de client de Tiendeo a Espanya podria ser Carrefour, ja que paga per tal que els seus catàlegs hi apareixin a la pàgina principal de Tiendeo (també anomenada

landing) o que en les cerques dels usuaris, els seus articles o sales hi apareguin en les primeres posicions de la llista.
És un actor que s'haurà de tenir en compte a l'hora d'ordenar la llista de hits d'un usuari, perquè un article pot trobar-se en campanya (un client ha pagat per promocionar aquell article o sale) i per tant haurà d'estar posicionat millor que un altre que no ho estigui.

- Tiendeo. Aquest serà el principal beneficiari del treball realitzat, perquè obtindrà millors resultats en les seves cerques i per tant, idíl·licament, sempre que es busqués alguna cosa, l'usuari ho trobaria i podria provocar que hi cliqués en articles o sales les quals es troben en campanya, generant que, si aquest article o sale fos d'un nou client, aquest passés a ser un client habitual pels bons resultats, a causa del nombre elevat de visites en els seus productes promocionats.

- Els desenvolupadors. Seran els encarregats de dirigir i desenvolupar el projecte per tal que es pugui realitzar i finalitzar.

2. Justificació del projecte

L'objectiu del projecte és poder crear un cercador per a Tiendeo. Més exactament, se li afegeix la funció de poder cercar qualsevol classe de text, cosa que actualment és incapaç de fer. Si fos el cas que l'usuari hi cerqués una paraula que es troba precalculada, se li retornaria la llista precalculada, però amb l'ordenació del nou search engine a desenvolupar. S'entrarà en més detall sobre el cercador actual i la necessitat del seu canvi en el subapartat 2.2. Selecció de programari.

El disseny d'un motor de cerca és una àrea la qual és molt tractada, ja que, com s'ha parlat anteriorment, la quantitat de cerques en els darrers anys ha crescut exponencialment i la competència també. La majoria d'organitzacions que treballen en la web requereixen d'un si estan enfocats als usuaris.

Per tant, es passarà a veure aquestes solucions i si són aprofitables per aquest projecte. Seguidament, es veurà el programari que utilitza Tiendeo pel cercador actual i si és necessari canviar-lo o continuar amb aquest.

2.1. Estudi de les solucions existents

Els cercadors generalment són eines desenvolupades *in-house*[2], ja que cada organització té les seves pròpies dades privades amb les quals cercar i tractar. És a dir, no hi ha cap projecte *open-source*[3] el qual sigui un cercador per a *e-commerce*[4] com és Tiendeo.

Tot i això, tota organització que vol desenvolupar un cercador es troba davant de dues opcions amb les quals treballar:

- Utilitzar search engines ja desenvolupats i configurar-lo i adaptar-lo a les dades per l'organització.
- Desenvolupar un search engine des de zero. Implica utilitzar algoritmes tals com TF-IDF[4], paral·lelitzant entre els milions de dades que tens de la forma més eficient possible.

Clarament, la millor opció és la primera, perquè només t'hi has de centrar a fer funcionar el cercador, inserint les dades pròpies de l'organització d'una forma adient i configurar-lo de la millor forma possible sense la necessitat d'haver-se de preocupar de si el search engine desenvolupat i el temps d'executar una *query*[5] davant de milions de documents no és prou ràpid. Igualment, l'usuari que fa ús del

[2]Eines desenvolupades dins d'una organització
[3]És un tipus de llicència de software que garanteix els drets d'ús lliure.
[4] Comerç electrònic
[5] Interacció a realitzar per extreure informació de documents.

search engine s'haurà de preocupar de fer les cerques adients per garantir que el temps en la cerca és bo.

Hi ha de search engines open-source tals com Elasticsearch, Solr, Sphinx… i d'altres que no ho són com Splunk.
La majoria d'organitzacions utilitzen Elasticsearch com es pot observar en la figura 4, ja que garanteix un molt bon temps per cerca, documentació ben detallada, constant manteniment, bona escalabilitat, fàcil ús i una llicència lliure.

Rank			DBMS	Database Model	Score		
Sep 2021	Aug 2021	Sep 2020			Sep 2021	Aug 2021	Sep 2020
1.	1.	1.	Elasticsearch	Search engine, Multi-model	160.24	+3.16	+9.74
2.	2.	2.	Splunk	Search engine	91.61	+1.01	+3.71
3.	3.	3.	Solr	Search engine, Multi-model	49.81	-1.26	-1.81
4.	4.	4.	MarkLogic	Multi-model	9.60	+0.45	-2.34
5.	5.	↑ 7.	Sphinx	Search engine	7.64	-0.11	+1.17
6.	6.	↓ 5.	Algolia	Search engine	7.30	+0.24	+0.53
7.	7.	↓ 6.	Microsoft Azure Search	Search engine	6.85	+0.71	+0.15
8.	8.	8.	ArangoDB	Multi-model	4.79	+0.53	-1.01
9.	9.	↑ 10.	Virtuoso	Multi-model	4.42	+0.19	+1.86
10.	10.	↓ 9.	Amazon CloudSearch	Search engine	2.20	+0.02	-0.41

Figura 4: Mostra el top 10 Search Engines del mercat.
Font: db-engines.com/en/ranking/search+engine

Tot i això, abans de definir el programari, s'ha de centrar el focus en quins estris utilitza Tiendeo per les seves cerques actuals i si és necessari canviar-los dràsticament per utilitzar un search engine d'aquests últims mostrats o adaptar-lo.

2.2. Selecció de programari

El cercador actual està programat en *Python*, ja que és un dels millors llenguatges de programació pel tractament de grans quantitats de dades.
A més, s'utilitzen una sèrie de *pipelines*[6] potenciades per *Kubernetes*[7] i *Docker*[8] i estan desenvolupades per ells. Amb aquestes pipelines s'aconsegueix, executant només un fitxer, un procés per fases de l'estil:

1. Extreure d'*AWS*[9] les dades necessàries.
2. Modificar com convingui totes les dades extretes.
3. Pujar a AWS les noves dades transformades.

[6]Conjunts d'elements de processament de dades connectats en sèrie, on la sortida d'un element és l'entrada del següent[5].
[7] Open-source software que permet el maneig de dades en contenidors.
[8] Projecte open-source que automatitza les aplicacions emmagatzemant-les en contenidors.
[9]Amazon Web Services. És un servei en cloud per a poder emmagatzemar dades.

Si s'observa com és el disseny del cercador actual amb detall, seria:

1. Agafar totes les paraules a precalcular (productes, retailers, marques…)
2. Calcular la intersecció entre totes aquestes paraules a precalcular amb la seva etiqueta corresponent. És a dir, si s'està iterant per tots els productes, s'agafa, de la base de dades, tots els articles etiquetats pel producte de la iteració corresponent. Així, es tindrà una llista, on, a cada posició, hi haurà una tupla paraula-llista de resultats.
3. Per a cada llista de resultats de la llista, es reordena aquesta a partir d'una puntuació que cada article o sale té i és individual. Aquesta puntuació és un nombre que es calcula amb diferents factors, tals com si es troba en campanya, ha sigut publicat recentment, etc.

Com es pot veure, aquest cercador no és adaptable si es vol fer cerques per a qualsevol entrada, perquè si un usuari introdueix un text el qual no correspon amb un producte, marca, retailer, etc., la intersecció entre el text i els articles o sales donarà buida, i, per tant, la cerca no donaria resultats. A més, si el tagueig[10] del producte d'un article és erroni, aquest article sortirà en una llista a la qual no pertany.

Per tant, la solució proposada (i analitzada i justificada a l'apartat 8. Anàlisi de diferents tecnologies) és aprofitar el disseny de pipelines en **Kubernetes** per a fer les transformacions necessàries de les dades i pujar-les al search engine. No hi ha cap necessitat de migrar a un altre llenguatge de programació que no sigui **Python**, ja que, a part de ser el llenguatge per excel·lència en el tractament de dades per la gran facilitat i nombre de llibreries que ofereix, el search engine escollit, el qual és **Elasticsearch**, suporta aquest llenguatge. A més, per a poder fer les transformacions de la gran quantitat de dades amb Python de la forma més ràpida possible, s'utilitzarà la llibreria **Pandas.**

[10] Acte de posar un valor en un lloc en específic.

3. Abast del projecte

En aquest apartat es parlarà de l'abast del projecte. Es tractaran els objectius i els subobjectius, requeriments i possibles obstacles i riscos que hi poden sorgir.

3.1. Objectius

L'objectiu genèric principal d'aquest projecte és el desenvolupament i posterior anàlisi d'un cercador que sigui capaç de fer *full-text search*[11].

El desenvolupament consistirà a fer el tractament de dades necessari i configurar com esdevingui el search engine per tal que sigui capaç de fer les cerques. La posterior anàlisi dirà si el cercador està funcionant correctament o no (una possible manera de veure-ho és si la gent que cerca per buscar articles o sales acaba trobant el que busca) i si el servidor configurat és capaç de poder suportar la càrrega de les cerques que efectuaran els usuaris.

A continuació, es tenen els següents subobjectius del treball:

- Rendiment. Es necessita que qualsevol cerca d'un usuari sigui inferior a 100 ms aproximadament i que els processos remots encarregats d'actualitzar les dades a Elasticsearch no triguin més de 5 minuts per país.
- Investigar i desenvolupar característiques per a poder maximitzar els bons resultats en qualsevol mena de cerca.
- Estudi i anàlisi de les interaccions dels usuaris amb el cercador, és a dir, veure l'índex d'obertures d'articles o sales per cada cerca.
- Anàlisi de la bona configuració del cercador per a cada país. S'ha de configurar el cercador per més de 20 llengües diferents i s'ha d'aconseguir tenir un índex d'obertures semblants entre tots els països.
- Analitzar la necessitat de canviar d'Elasticsearch a una altra tecnologia.
- Desenvolupar un suggeridor pel cercador. Si es posa d'exemple Google, quan s'escriu una paraula, aquest et mostra una llista de possibles paraules autocompletades que són les més cercades del moment. Aquesta és una funcionalitat que es vol intentar aplicar.

3.2. Requisits

3.2.1. Requisits funcionals

Un requisit funcional que necessita el cercador és permetre a l'usuari poder cercar per a qualsevol paraula i que li retorni una llista de resultats corresponent a la cerca

[11] Poder fer cerques amb qualsevol tipus de text com entrada.

que ha efectuat. És a dir, que el cercador tingui l'habilitat per identificar els documents més significatius davant d'una cerca.
També seria necessari poder fer suggeriments sobre paraules mal escrites als usuaris.

3.2.2. Requisits no funcionals

Pel que fa als requeriments els quals no són necessaris directament per tal que el cercador funcioni són:

- Escalabilitat. El cercador hauria de ser força independent sobre la quantitat de les dades per les quals se cerca.
- Adaptabilitat. Hauria de ser prou adaptable per si es volgués poder cercar altres entitats que no siguin ni articles ni sales.
- Hauria de tenir un índex d'obertura per cerca superior al 30%.
- La disponibilitat del cercador hauria de ser, idíl·licament, del 100%.
- En el cas que s'ajuntessin cerques massives d'usuaris i processos remots per actualitzar dades, s'hauria de donar prioritat a les cerques d'usuaris.
- Com s'ha esmentat prèviament, el temps mitjà per cerca no ha de sobrepassar els 100 ms. Si no es complís, el posicionament de Tiendeo a Google es veuria disminuït.

3.3. Obstacles i riscos

Els principals obstacles que poden ocórrer en aquest treball són els següents:

- Inexperiència en Elasticsearch. Aquest search engine ha estat en desenvolupament durant més de deu anys i la quantitat de funcionalitats que té són molt grans. Per tant, fins a tenir una base del seu funcionament, pot provocar l'endarreriment del desenvolupament del projecte.
- Desconeixement de l'arquitectura. En ser una empresa amb deu anys de desenvolupament, el desconeixement d'aquesta arquitectura pot provocar que els primers dies el treball efectiu sigui reduït considerablement.

Com en qualsevol projecte de certa magnitud, poden ocórrer certs riscos els quals ens hem de preparar amb antelació i són els que es mostren a continuació:

- La possibilitat de poder perdre dades del desenvolupament del treball. Per intentar evitar aquest factor de risc, s'utilitzarà GitLab (com es comenta en l'apartat 4.1. Eines de desenvolupament).
- La disponibilitat dels servidors d'Elasticsearch. Per cada usuari que hi faci una cerca que no està precalculada, s'haurà d'accedir a un dels servidors

configurats d'Elasticsearch. Si en un moment donat hi hagués molt tràfic en la web o en l'app i se sumessin els processos remots per actualitzar les dades de nous articles i/o sales, podria fer col·lapsar algun servidor. A més, si aquests processos remots fossin massa llargs, incrementaria el risc de caiguda dels servidors, pel fet que a més temps s'estigui operant en el servidor, més marge de temps hi ha per un possible augment de tràfic (per això ens hem posat de subjectiu el rendiment).

4. Metodologia i rigor

Tiendeo és una empresa que es troba en constant desenvolupament i provoca que hi puguin haver canvis en les dades amb les quals es tracten pel search engine. Per tant, ha de ser flexible a possibles canvis.

A més, el fet d'entrar de nou en una empresa amb un software molt desenvolupat sense conèixer res de l'arquitectura d'aquest, implica que hagi de ser un tipus de metodologia en què el coneixement previ no sigui necessari i es pugui anar definint de manera dinàmica el progrés que s'ha anat aconseguint.

Llavors, la millor metodologia per aquest treball és la Scrum, ja que és perfecte per projectes on els integrants no tenen coneixements en començar el projecte i adequada si és complex.
Per seguir amb la metodologia Scrum, el seguiment del treball serà mitjançant reunions diàries d'una duració màxima de 15 minuts on es parlarà del treball realitzat el dia previ a la reunió i el que es vol aconseguir en el mateix dia o si ha sorgit algun obstacle.

El desenvolupament del projecte es durà a terme dins les oficines de Tiendeo o tele-treballant amb una VPN. Els dies que es pugui treballar en les oficines i hi hagi sales de reunió disponibles, les reunions es duran a terme en aquestes.

4.1. Eines de desenvolupament

Per facilitar el seguiment dels objectius, s'utilitzarà *Trello*[6], on és una eina que, via online, es poden establir tasques i dates de venciment d'aquestes.

També, per tenir un control de versions del projecte, s'utilitzarà *GitLab*[7], ja que Tiendeo té un servidor propi on pugen qualsevol canvi en el desenvolupament de noves característiques.

I, pel desenvolupament de codi, s'utilitzarà *Visual Studio Code*[8], ja que és una aplicació perfecta pel desenvolupament de software, ja que té eines per poder depurar codi, pujar canvis ràpidament a GitLab, és open-source i aporta extensions per fer el codi més llegible.

4.2. Mètode de validació

De cara al mètode de validació, en les reunions diàries s'avaluarà l'assoliment dels objectius proposats a curt termini i parlar les properes millores. I, si s'ha acabat alguna nova característica, es faran petites proves a països amb poc tràfic per

provar-les. Amb aquestes proves es veuran més fàcilment possibles errors o problemes que sense la interacció d'usuaris no es podrien veure.

5. Planificació temporal

Tot el projecte té una durada aproximada d'uns sis mesos, començant el dia 21 de juny del 2021 i acabant, aproximadament, el desembre del 2021. Es treballarà cada dia d'entre setmana durant 7 hores, fet que la durada en hores prevista aproximada desenvolupant el treball sigui d'unes 1000.

A l'etapa final, també es farà paral·lelament la documentació del projecte, ja que el torn de lectura està previst per la setmana del 24 de gener del 2022, fet que fa tenir marge aproximadament d'un mes per a poder detallar i finalitzar el projecte, així com poder enfocar-se en la presentació final.

5.1. Descripció de les tasques

Per tal de poder finalitzar a temps el TFG en aquesta fita final aproximada, es durà a terme una divisió de tasques en 4 fases: gestió del projecte, anàlisi de la millor tecnologia a agafar, desenvolupament del cercador i anàlisi de resultats. Per a cada tasca que es descriurà en cada fase, s'especificarà les seves dependències, la seva estimació en hores i el seu ID de tasca per després fer referències a aquestes d'una forma més senzilla.

5.1.1. Gestió del projecte

Aquesta fase inclou tasques essencials, tals com la planificació i organització del projecte i reunions per a la validació d'aquest.

- **GP1. Context i abast (20 hores):** Es definirà el context i l'abast que tindrà el projecte. Dependències: No hi ha dependències per a aquesta tasca.
- **GP2. Planificació temporal (15 hores):** Es du a terme la planificació temporal, diagrama de Gantt i la gestió dels riscos. Dependències: GP1.
- **GP3. Gestió econòmica i sostenibilitat (10 hores):** Es redacta el pressupost i un informe de sostenibilitat sobre el projecte. Dependències: GP2.
- **GP4. Documentació de la fase inicial (10 hores):** Es fa un recull dels tres punts anteriors i es fa una redacció i revisió final. Dependències: GP3.
- **GP5. Documentació de la fase de seguiment (30 hores):** Es prepara l'entrega, redactant l'estat i el progrés del projecte. Dependències: GP4.
- **GP6. Documentació de la fase final (60 hores):** Es documenta la part final del projecte abans de l'entrega. Dependències: GP5.
- **GP7. Preparació de la defensa (30 hores):** Preparació de la presentació del projecte davant el tribunal. Dependències: GP6.

- **GP8. Daily Scrum (15 minuts diaris):** Reunió diària entre setmana amb l'equip per comentar els avanços i propers passos. Dependències: No hi ha dependències per a aquesta tasca.

5.1.2. Anàlisi de tecnologies

Aquesta fase consisteix en la selecció de la millor tecnologia (search engine) per a poder fer el cercador i el seu respectiu aprenentatge. Com s'ha explicat en apartats anteriors, s'ha de tenir en compte que Elasticsearch és una tecnologia ja implantada en l'empresa i si s'hagués de fer un canvi, hauria de ser una tecnologia clarament millor.

- **AT1. Cerca de fonts d'informació (28 hores)**: Es fa una investigació sobre fonts d'informació de les principals tecnologies del mercat i les seves característiques. Dependències: No hi ha dependències per a aquesta tasca.
- **AT2. Anàlisi de les principals tecnologies (35 hores):** Es du a terme un estudi sobre les característiques principals de les diferents tecnologies i s'escull la més adient. Dependències: AT1.
- **AT3. Aprenentatge de la tecnologia (49 hores):** Es fa un curset i cerca de la informació adient per tal d'aprendre una base per a poder treballar amb aquesta tecnologia. Dependències: AT2.
- **AT4. Anàlisi per a l'elecció de servidors (21 hores):** Es farà una anàlisi per a escollir els servidors amb les característiques més adients pel search engine. Dependències: AT3.

5.1.3. Desenvolupament del cercador

En aquesta fase es comença amb el desenvolupament del cercador, seleccionant les dades més adients per a pujar al search engine, fer una query adient i extreure les dades.

- **D1. Cerca de les principals fonts de dades (28 hores):** Per tal d'obtenir el text que els usuaris poden cercar, es fa una cerca en les bases de dades de Tiendeo. Dependències: No hi ha dependències per a aquesta tasca, ja que el text en el qual se cerca és independent del search engine a utilitzar.
- **D2. Extracció de les dades (21 hores):** Fer pipelines que, executades en un servidor remot, extreguin les dades necessàries pel cercador. Dependències: D1.
- **D3. Transformació de les dades (90 hores):** En les pipelines creades, es faran les transformacions necessàries pel cercador. Dependències: D2.
- **D4. Fer tests abans de pujar dades (28 hores):** Abans de pujar dades al search engine i a les bases de dades de Tiendeo, aquestes hi hauran de

passar uns tests automàtics tals com: la mida màxima dels fitxers, l'estil de les dades, etc. Dependències: D3.

- **D5. Configurar el search engine (70 hores):** Com cada país té la seva pròpia llengua, s'haurà de cercar documentació oficial i configurar de manera específica cada idioma dins dels que tracta Tiendeo. Dependències: AT4, D4.
- **D6. Pujada de les dades al search engine (40 hores):** Aprofitant les pipelines anteriors, s'amplien per afegir la pujada d'aquestes dades al search engine. Dependències: D4, D5.
- **D7. Creació de les queries (90 hores):** Es durà a terme una cerca en la documentació oficial del search engine sobre les característiques que es poden aplicar per tal d'extreure la informació de la millor manera possible. Es faran queries diferents, depenent si es fa una cerca precalculada o no. Tot i això, les queries entre elles seran semblants. Dependències: AT3, D6. Ja que per a poder començar a veure com formar una query, s'ha hagut de seleccionar el search engine i tenir coneixement sobre aquest, a més de poder fer proves amb les dades pujades.
- **D8. Paraules precalculades (77 hores):** Es faran les pipelines necessàries per a crear les llistes de tuples paraula-llista de resultats per cada país, utilitzant les queries ja creades. Dependències: D7. Per tenir la llista de resultats de cada paraula, s'ha de tenir la query per a extreure informació dels documents.
- **D9. Creació d'*endpoints*[12] (35 hores):** Es faran endpoints en l'*API*[13] de DATA Profiling de Tiendeo per tal que els encarregats a la web consumeixin les noves ordenacions dels articles i/o sales pel cercador. Dependències: D8.
- **D10. Autocompletat del cercador (100 hores):** Es desenvoluparà un suggeridor/autocompletat de paraules. Dependències: D6.

5.1.4. Anàlisi de resultats

Un cop desenvolupat el cercador i establert a cada país per tal que tot usuari pugui utilitzar-lo, es duran a terme les següents tasques per tal de poder millorar-lo. Totes les tasques d'aquesta fase són dependents de l'equip de WEB, com es veu amb més detall al diagrama de Gantt (figura 5) i s'explica a l'apartat 5.4. Gestió de risc: Plans alternatius i obstacles.

- **AR1. Anàlisi dels índexs d'obertures (56 hores):** Per a cada país, es farà una anàlisi exhaustiva per si les obertures dels usuaris han baixat i/o pujat. A més, s'aprofitaran els articles i sales que són més oberts per tal de posicionar-los millor. Dependències: D9. S'ha de tenir acabat el cercador.

[12] És una forma per a poder interaccionar amb el software que l'API gestiona.
[13] Application Programming Interface. És un conjunt de rutines que donen accés a diferents funcions d'un software [9].

- **AR2. Anàlisi dels no-results (35 hores):** S'analitzaran les causes de les cerques les quals no obtenen resultats. Dependències: D9.
- **AR3. Comparació de resultats amb el cercador anterior (49 hores):** Es provarà d'analitzar la qualitat de les cerques del cercador anterior amb el nou desenvolupat. Clarament, només es compararan les cerques que poden ser precalculades, ja que l'anterior cercador no té la capacitat per cercar qualsevol mena d'entrada. Dependències: D8.
- **AR4. Anàlisi d'escalabilitat (56 hores):** Es farà una anàlisi sobre si la capacitat dels servidors del search engine és suficient per poder aguantar les cerques dels usuaris, l'autocompletat, etc. Dependències: D9.

5.2. Recursos

En el projecte seran necessaris recursos tant humans, materials com indirectes per a dur a terme les tasques proposades anteriorment.

5.2.1. Recursos humans

Faran falta els següents recursos humans:

- **L'equip de DATA:** L'equip on l'autor, el director i codirector del projecte es troben involucrats. Serà necessari per a desenvolupar el projecte, consultar dubtes i solucions, etc. A més, a l'hora de fer la daily scrum, com està present tot l'equip, sempre es poden tenir idees noves i/o diferents.
- **L'equip de SYSTEMS:** Per tal de detallar si les característiques donades per establir nous servidors són suficients i per posar-los en marxa.
- **L'equip de WEB:** Serà l'encarregat de canviar l'ordenació que és utilitzada pel cercador actual pel nou a desenvolupar.
- **L'equip de SEO:** Per consultar dubtes que hi puguin sorgir davant les dades que s'hauran d'utilitzar.
- **El ponent del projecte:** Pere Barlet Ros també donarà suport per possibles dubtes i ajudant quan sigui necessari.

5.2.2. Recursos materials

Pel que fa als recursos materials, s'utilitzen:

- **Atenea:** Web d'on treure la informació necessària per documentar el projecte.
- **AWS:** Per a consultar dades i pujar dades al servei d'amazon S3.
- **Eines de Google:** Google Docs, Google Meet, Google Calendar, Google Chrome, Google Spreadsheets, Google Chat i Gmail.
- **Elasticsearch:** Tecnologia open-source utilitzada pel desenvolupament del cercador.

- **Gantt Project:** Eina per fer el diagrama de Gantt.
- **Gitlab i SourceTree:** Per tenir control de versions del projecte.
- **Jenkins:** Automatització dels CI[14] servers.
- **Kubernetes i Docker:** Eina per a poder posar en producció els contenidors (les pipelines fetes).
- **Ordinador portàtil** amb processador Intel(R) Core(TM) i7-8850H, 16GB de RAM, 512GB (HDD) de memòria i sistema operatiu Windows 10 i Ubuntu.
- **Perifèrics:** Ratolí, teclat i monitor.
- **Servidors:** El search engine necessitarà servidors per poder respondre a totes les peticions dels usuaris.
- **Trello:** Eina per tenir seguiment de les tasques a fer.
- **Visual Studio Code:** IDE per desenvolupar el projecte.

5.2.3. Recursos indirectes

Per últim, es troben els recursos indirectes, que són els que tenen a veure amb l'espai de treball i són els següents:

- **Materials per poder treballar:** Taula i cadira.
- **Necessitats bàsiques:** Transport, electricitat i internet.
- **Lloc de treball:** Dos de cada cinc dies es realitza teletreball (en el mateix domicili) i els restants (⅗) a les oficines de Tiendeo.

5.3. Estimació temporal

A continuació, en la taula 1, es té un resum amb l'estimació en hores per cada tasca i les seves respectives dependències.

ID	Tasca	Hores	Dependències
GP1	Context i abast	20	-
GP2	Planificació temporal	15	GP1
GP3	Gestió econòmica i sostenibilitat	10	GP2
GP4	Documentació de la fase inicial	10	GP3
GP5	Documentació de la fase de seguiment	30	GP4
GP6	Documentació de la fase final	60	GP5

[14] Continuous Integration. Administra el repositori compartit i actua com un árbit pel codi que ingresa [10].

GP7	Preparació de la defensa	30	GP6
GP8	Daily Scrum	0.25	-
AT1	Cerca de fonts d'informació	28	-
AT2	Anàlisi de les principals tecnologies	35	AT1
AT3	Aprenentatge de la tecnologia	49	AT2
AT4	Anàlisi per a l'elecció de servidors	21	AT3
D1	Cerca de les principals fonts de dades	28	-
D2	Extracció de les dades	21	D1
D3	Transformació de les dades	90	D2
D4	Fer tests abans de pujar dades	28	D3
D5	Configurar el search engine	70	AT4, D4
D6	Pujada de les dades al search engine	40	D4, D5
D7	Creació de les queries	90	AT3, D6
D8	Paraules precalculades	77	D7
D9	Creació d'endpoints	35	D8
D10	Autocompletat del cercador	100	D6, D10
AR1	Anàlisi dels índexs d'obertures	56	D9
AR2	Anàlisi dels no-results	35	D9
AR3	Comparació de resultats amb el cercador anterior	49	D8
AR4	Anàlisi d'escalabilitat	56	D9

Taula 1: Tasques a realitzar.
Font: Elaboració pròpia.

5.3.1. Planificació inicial

També tenim el diagrama de Gantt en la figura 5, el qual mostra la hipotètica planificació inicial on cada color serà una fase diferent. No es mostren les reunions diàries de 15 minuts per part de l'equip, ja que provoca que el diagrama no es vegi clar.

També es va preveure que la integració del cercador per part de la web trigués unes tres setmanes (és la tasca de color blanc que es veu al diagrama, en les primeres setmanes de novembre), per tant, s'aprofitaria aquell temps per fer tasques d'anàlisi i millorar el cercador, per això la sobreestimació en temps d'aquestes (s'entrarà en més detall a l'apartat 5.4. Gestió de risc: Plans alternatius i obstacles).

Nom	Inici	Finalització
Context i abast	22/9/21	30/9/21
Planificació temporal	1/10/21	8/10/21
Gestió econòmica i sostenibilitat	11/10/21	18/10/21
Documentació de la fase inicial	19/10/21	26/10/21
Documentació de la fase de seguiment	27/10/21	14/12/21
Documentació de la fase final	15/12/21	3/1/22
Preparació de la defensa	4/1/22	17/1/22
Cerca de fonts d'informació	21/6/21	24/6/21
Anàlisi de les principals tecnologies	25/6/21	1/7/21
Aprenentatge de la tecnologia	2/7/21	12/7/21
Anàlisi per a l'elecció de servidors	13/7/21	15/7/21
Cerca de les principals fonts de dades	21/7/21	26/7/21
Extracció de les dades	27/7/21	29/7/21
Transformació de les dades	30/7/21	17/8/21
Fer tests abans de pujar dades	18/8/21	23/8/21
Configurar el search engine	24/8/21	6/9/21
Pujada de les dades al search engine	7/9/21	14/9/21
Creació de les queries	15/9/21	1/10/21
Paraules precalculades	4/10/21	18/10/21
Creació d'endpoints	19/10/21	25/10/21
Intregació a web	26/10/21	15/11/21
Autocompletat del cercador	2/12/21	22/12/21
Anàlisi dels índexs d'obertures	26/10/21	4/11/21
Anàlisi dels no-results	16/11/21	22/11/21
Comparació de resultats amb el cercador anterior	5/11/21	15/11/21
Anàlisi d'escalabilitat	16/11/21	1/12/21

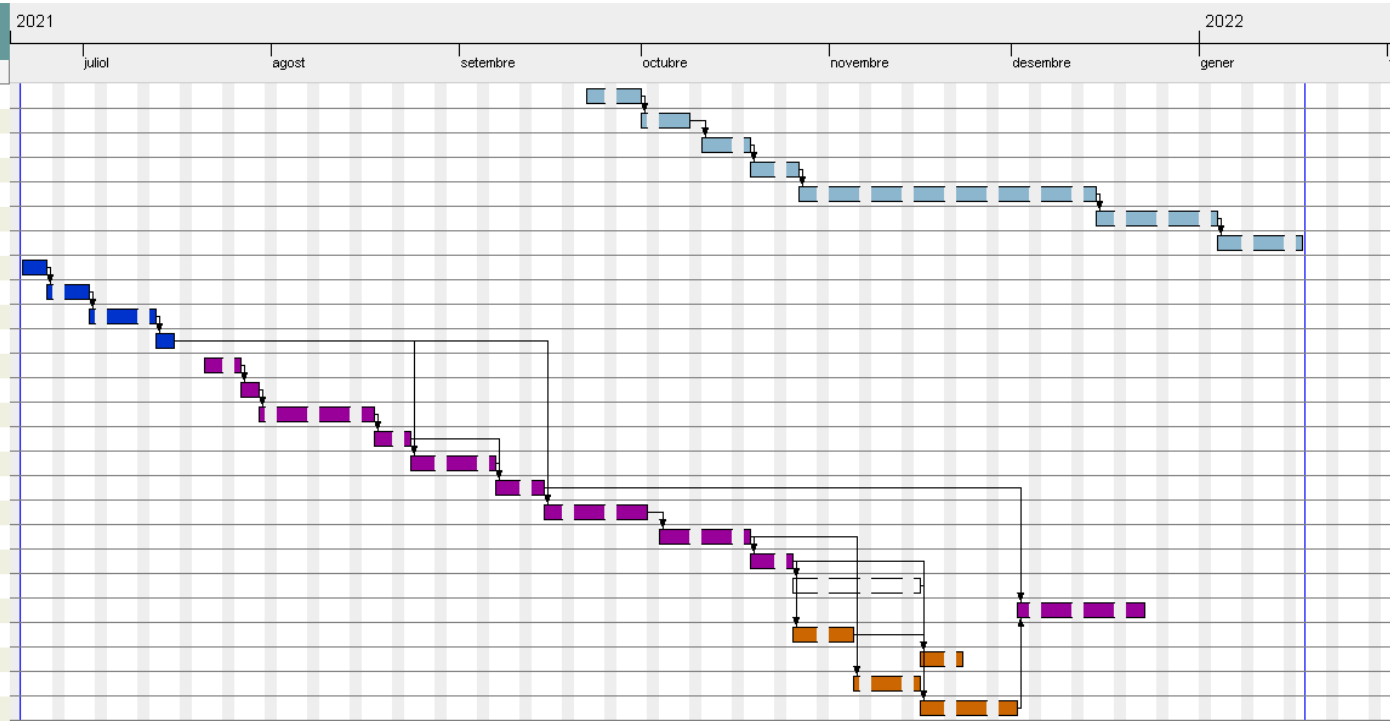


Figura 5: Diagrama de Gantt de les tasques a realitzar.
Font: Elaboració pròpia.

5.3.2. Planificació final

La planificació final del projecte és igual a la inicial feta, tret que òbviament hi ha hagut tasques que el seu temps estimat no ha sigut exacte, tot i que s'ha aproximat bastant. A la planificació inicial, s'havien sobreestimat algunes tasques (com bé es justifiquen a Gestió de risc: Plans alternatius i obstacles) i aquesta sobreestimació ha sigut la correcta, ja que el temps trigat ha sigut el sobreestimat.

Nom	Inici	Finalització
Context i abast	22/9/21	30/9/21
Planificació temporal	1/10/21	8/10/21
Gestió econòmica i sostenibilitat	11/10/21	18/10/21
Documentació de la fase inicial	19/10/21	26/10/21
Documentació de la fase de seguiment	27/10/21	14/12/21
Documentació de la fase final	15/12/21	3/1/22
Preparació de la defensa	4/1/22	17/1/22
Cerca de fonts d'informació	21/6/21	24/6/21
Anàlisi de les principals tecnologies	25/6/21	1/7/21
Aprenentatge de la tecnologia	2/7/21	12/7/21
Anàlisi per a l'elecció de servidors	13/7/21	15/7/21
Cerca de les principals fonts de dades	21/7/21	26/7/21
Extracció de les dades	27/7/21	29/7/21
Transformació de les dades	30/7/21	17/8/21
Fer tests abans de pujar dades	18/8/21	23/8/21
Configurar el search engine	24/8/21	6/9/21
Pujada de les dades al search engine	7/9/21	14/9/21
Creació de les queries	15/9/21	1/10/21
Paraules precalculades	4/10/21	18/10/21
Creació d'endpoints	19/10/21	25/10/21
Intregació a web	26/10/21	15/11/21
Autocompletat del cercador	26/10/21	15/11/21
Anàlisi dels índexs d'obertures	16/11/21	25/11/21
Anàlisi dels no-results	7/12/21	13/12/21
Comparació de resultats amb el cercador anterior	26/11/21	6/12/21
Anàlisi d'escalabilitat	7/12/21	22/12/21

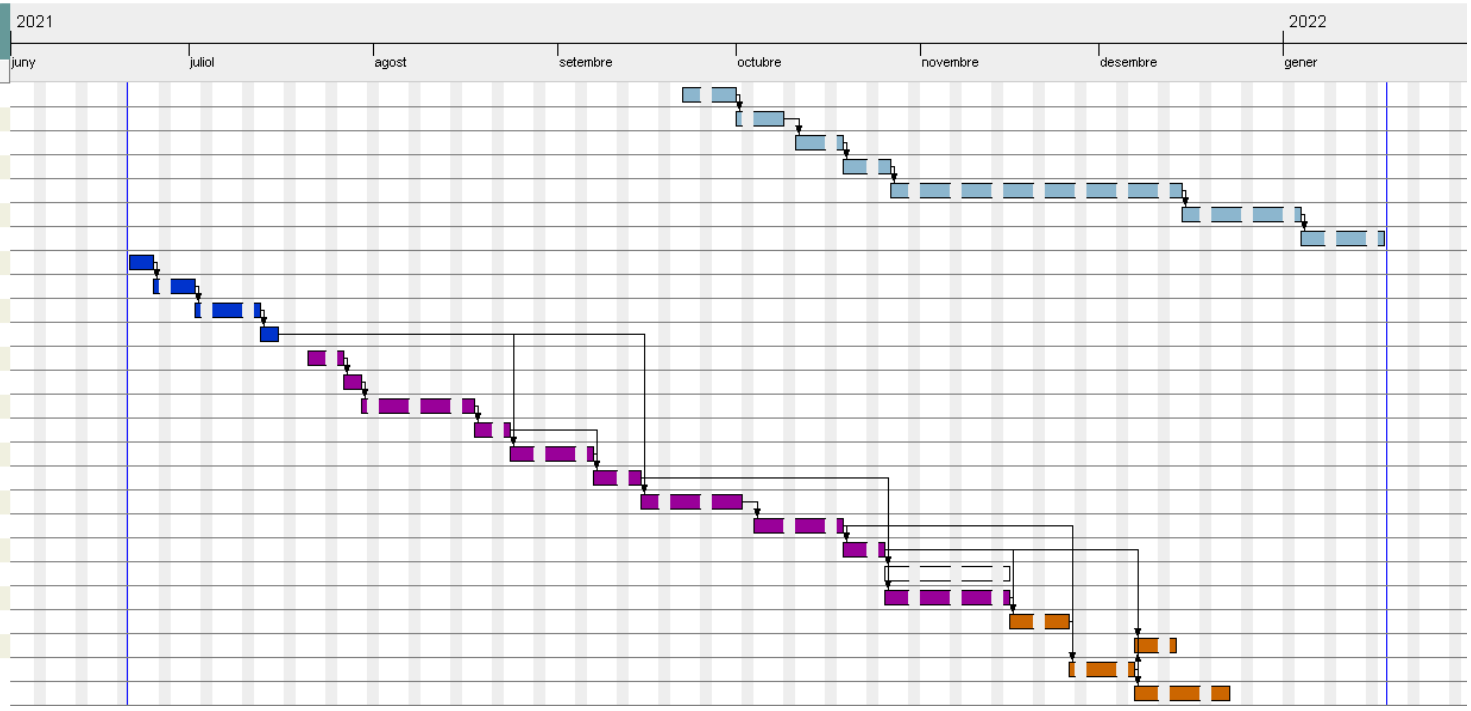


Figura 6: Mostra la planificació final del projecte.
Font: Elaboració pròpia.

Tot i això, com es pot observar en la figura 6, l'únic que s'ha modificat en la planificació és un petit canvi en prioritats. És a dir, en comptes de deixar cap al final l'autocompletat, s'ha realitzat el seu desenvolupament just quan l'equip de web havia de posar el cercador a la web. Com es comenta al següent subapartat, aquesta tasca havia estat sobreestimada i posada al final per si sorgien imprevistos i no donés temps a acabar-la. Però, com fins aquell punt no havia sorgit cap imprevist i va agafar més importància el seu desenvolupament, va ser prioritzada. A més, aquesta, va ser més llarga de l'esperat i es va acabar trigant el mateix que el temps estipulat a la planificació inicial, el qual estava sobreestimat.

A més, s'ha aprofitat el temps de les tasques d'anàlisi (l'última fase) i, amb els resultats que s'anaven obtenint, s'anaven fent petites millores en el cercador. Llavors, en fer sobreestimació en aquestes tasques, el temps dedicat ha estat l'estipulat.

Per tant, el pressupost es quedarà igual, ja que el temps emprat, potser no ha estat exacte al de la planificació inicial, però sí molt semblant, ja que la majoria de tasques s'han realitzat en acord amb la planificació i el temps estipulat era el correcte.

5.4. Gestió de risc: Plans alternatius i obstacles

Com s'ha parlat anteriorment a l'apartat 3.3. Obstacles i riscos, el projecte té certs riscos i alguns obstacles que poden fer enrederir l'entrega d'aquest. Per tant, en aquest apartat s'explicaran les mesures que se'n van prendre per gestionar possibles riscos que podien sorgir.

Primerament, es té el risc de poder perdre dades a mesura que es desenvolupa el treball, la seva probabilitat és molt baixa, ja que s'està utilitzant Gitlab per evitar aquest risc.

També existeix el risc que la disponibilitat dels servidors del search engine sigui baixa per l'afluència d'usuaris. La probabilitat que succeeixi és moderada, però l'impacte d'aquest és molt elevat, ja que deixaria de funcionar el cercador per moments, i per tant, s'haurien de reiniciar aquests. Llavors, s'haurien d'afegir algunes tasques alternatives d'anàlisi sobre la quantitat de servidors que es necessiten per sumar-los als actuals (tasca AR4). Això provocaria que es necessitin més servidors en cas que sigui necessari (més recursos materials).

Seguidament, es té l'obstacle de la inexperiència en Elasticsearch i el desconeixement de l'arquitectura. La probabilitat d'aquestes és elevada i l'impacte també, per això, les tasques inicials de la fase de desenvolupament del cercador i la tasca d'aprenentatge de la tecnologia s'han elevat el seu nombre d'hores.

A més, també hi ha un obstacle que és la implementació del nou cercador a la web per part de l'equip de WEB (exactament per poder cercar qualsevol paraula, ja que les paraules precalculades haurien d'estar disponibles pels usuaris un cop acabada la tasca D9). Això provocarà una aturada del projecte d'unes dues setmanes fins que no estigui acabat. Com l'impacte és molt gran, se sobreestimarà el temps d'implementació a tres setmanes i les tasques que es desenvolupin en aquell període de temps també es veuran sobreestimades.

Per últim, existeix el risc que per possibles imprevistos, no hi hagi temps a desenvolupar l'autocompletat del cercador, per això, s'havia ficat com a última tasca en la planificació inicial i s'ha sobreestimat el seu temps. La probabilitat és moderada i l'impacte elevat.

6. Pressupost

En aquest apartat primerament es parlarà sobre quin és el pressupost del projecte, analitzant-lo. Seguidament, es parlarà de com es gestionarà aquest cost, tenint en compte possibles imprevistos.

6.1. Identificació dels costos

En tot projecte són necessaris una sèrie de recursos humans i materials, igual que recursos indirectes tal com l'espai de treball en el desenvolupament. A més, també s'han de tenir en compte costos per possibles imprevistos. Per tant, en aquest subapartat es parlarà sobre cada un d'aquests punts.

6.1.1. Recursos humans

Per tal d'assignar els costos del personal, primerament es llisten les persones directament involucrades en el projecte, és a dir, les necessàries per a poder dur a terme les tasques descrites en l'apartat 5.3. Descripció de les tasques, per rol. A més, se'ls hi assignarà un ID per tal de poder fer referències a aquests de manera simplificada:

- [PO] Product Owner (Director)
- [DTL] Data Tech Lead (Codirector)
- [DB] Developer becari (autor del projecte)
- [DBS] Developer backend senior (equip de web)
- [DFS] Developer frontend senior (equip de web)
- [SE] System Engineer (equip de systems)

A continuació, per cada rol, es mostra la mitjana dels sous en brut extrets de la pàgina *Glassdoor*[11]. Vegeu la taula 2, on es pot observar el salari per hora en brut per cada rol i el seu respectiu cost a l'empresa.

Rol	Sou/hora	Sou/hora + Seguretat Social
Product Owner	24.91€	32.39€
Data Tech Lead	24.27€	31.55€
Developer becari	9€	11.7€
Developer backend senior	20.47€	26.61€

Developer frontend senior	21.47€	27.91€
System Engineer	18.02€	23.43€

Taula 2: Sou brut per hora per cada rol.
Font: Elaboració pròpia

Tot seguit, s'estima el cost per cada tasca definida. Per fer-ho, es té en compte, del total d'hores en poder dur a terme una tasca, les hores contribuïdes per cada persona (rol ID) que ha participat directament en el desenvolupament de la tasca. També s'afegeixen les tasques WEB (implantació del nou search engine a la pàgina web) amb una estimació per fer la tasca d'unes 35 hores i SYS (posar en marxa els servidors requerits), amb una estimació de 14 hores. A més, el cost total per tasca està calculat tenint en compte el Sou/Hora + Seguretat Social. Vegeu la taula 3:

ID Tasca	**Hora / Tasca**	**Hores**						**Cost**
		PO	DTL	DB	DBS	DFS	SE	
GP1	20	0.5		19.5				244.35€
GP2	15	0.5	0.5	14				195.77€
GP3	10	0.5		9.5				127.35€
GP4	10	0.5		9.5				127.35€
GP5	30	0.5		29.5				361.35€
GP6	60	1		59				722.69€
GP7	30	1		29				371.69€
GP8	0.25(x105)	0.25	0.25	0.25				78.75€
AT1	28	1		27				348.29€
AT2	35	1	1	33				450.04€
AT3	49			49				573.3€
AT4	21	0.5	0.5	20				265.97€
D1	28		2	26				367.3€
D2	21		2	19				285.4€
D3	90		5	85				1152.25€
D4	28		5	23				426.85€

D5	70		4	66				898.4€
D6	40		3	37				527.55€
D7	90		5	85				1152.25€
D8	77		2	75				940.6€
D9	35		5	30				508.75€
D10	100		5	95				1269.25€
AR1	56	1	2	53				715.59€
AR2	35	1	1	33				120.04
AR3	49	3		46				635.37€
AR4	56		2	54				694.9€
WEB	35				15	20		957.35€
SYS	14						14	328.02€
Total	1210.75	38.25	71.25	1052.25	15	20	14	14846.77€

Taula 3: Mostra en detall el cost dels recursos humans del projecte.
Font: Elaboració pròpia

Com es pot observar a la taula 3, el cost total estimat pels recursos humans és de 14846.77€. S'ha de tenir en compte que hi ha tasques sobreestimades i que els sous són aproximats però no exactes.

6.1.2. Recursos materials

Pel que respecta als recursos materials utilitzats cada dia pel desenvolupament del projecte, es poden subdividir en dos: els relacionats amb el software i amb el hardware. A més, per tal de calcular l'amortització dels recursos necessaris, s'aplicarà la següent fórmula:

$$Amortització = Cost\ [€]\ *\ ús\ en\ el\ projecte\ [hores]\ /\ (Vida\ útil\ [anys]\ *\ ús\ per\ any [dies]\ *\ dedicació\ al\ dia\ [hores]\ )$$

Per la part dels recursos de hardware, tal com Hisenda permet, la vida útil de cada un dels recursos materials correspondrà a quatre anys:

Recursos de hardware	Cost	Ús per any	Dedicació al dia	Ús en el projecte	Amortització [€]
Dell Latitude 5591 (ordinador portàtil [12])	1594.99€	220	7	1210.75	313.50€
Dell Optical Mouse-MS116 (ratolí [13])	19€	220	7	1210.75	3.73€
Dell-KB216 (teclat [14])	18.92€	220	7	1210.75	3.72€
Samsung LS24A310NHUXEN (monitor [15])	104.99€	220	7	1210.75	20.64€
Servidors [16]	264€	365	24	2880	21.70€
TOTAL					363.29€

Taula 4: Mostra l'amortització de cada recurs material de hardware.
Font: Elaboració pròpia

Com s'aprecia a la taula 4, l'ús per any en alguns casos és 220, ja que correspon als dies feiners per any, mentre que els servidors és l'any complet, ja que no s'aturen en cap moment.

A més, l'ús en el projecte és de 1210.75 hores, com s'ha vist en la taula 3, corresponent al total d'hores de totes les tasques. Tot i això, en el cas dels servidors, correspon a 2880 hores, ja que, aquests, s'utilitzen durant 4 mesos, les 24 hores del dia (4 mesos × 30 dies / mes × 24 hores / dia =2880 hores). Per últim, el cost dels servidors és definit per: 3 servidors × 22 € / mes × 4 mesos =264€.

Pel que respecte al software, tots són gratuïts menys AWS S3, on es paga 0.02 € / GB[17] al mes i es preveu una transmissió de dades de 500GB al mes.

A més, Hisenda, en ser software, permet fer canvis cada dos anys, ja que es poden rebre actualitzacions d'aquests. Per tant, la seva vida útil serà de dos anys. Vegeu la taula 5:

Recurs software	Cost	Ús per any	Dedicació al dia	Ús en el projecte	Amortització [€]
AWS S3	240€	220	8	1210.75	82.55€

Taula 5: Mostra l'amortització del recursos de software AWS S3.
Font: Elaboració pròpia

Llavors, el cost d'aquest és definit per: 0.02 € / GB mes × 500 GB / mes × 12 mesos / any × 2 anys = 240 €.

Per tant, es té un cost total dels recursos materials de 363.29€ + 82.55€ =445.84€.

6.1.3. Recursos indirectes

Com a últim tipus de recurs, es tenen els indirectes. Anteriorment, en el subapartat 5.2.3. Recursos indirectes s'ha comentat que el lloc de treball és variant, ja que ⅖ dies és teletreball i ⅗ a les oficines de Tiendeo. Per tant, es tindrà en compte per proporció d'ús el cost total d'aquest espai de treball.

Per tal d'obtenir un cost mensual aproximat del lloc de teletreball, es busca una àrea de *coworking* per a una persona en una zona propera a la resident en les pàgines Meetbcn[18] i OneCoWork[19]. S'ha obtingut doncs un cost mensual de 70€.
Com l'ús total és de ⅖, el cost mensual resulta en: 70€ × ⅖ = 28€.

En el cas del cost de l'oficina, com no es pot aconseguir el cost exacte, s'extreu, d'aquestes mateixes pàgines, un cost de 292€ / mes. Oficines les quals estan obertes les 24 hores i tenen espais per reunions (com Tiendeo). Igual que s'ha fet amb el lloc de teletreball, l'ús total en aquest cas és de ⅗, per tant, el cost mensual resulta en 292€ × ⅗ = 175.2€.
A més, com l'oficina és compartida simultàniament per 55 persones, s'aplicarà la part proporcional que hi participa en el projecte (6 persones, com s'ha especificat en el subapartat 6.1.1. Recursos humans) del total, obtenint doncs un cost mensual de: 175.2€ × 6/55 = 19.11€.
Vegeu la taula 6 on es fa un resum:

Recursos indirectes	Cost mensual [€]	Temps d'ús	Cost total [€]
Teletreball	28€	6 mesos	168€
Oficina	19.11€	6 mesos	114.68€
TOTAL			282.68€

Taula 6: Mostra els costos de l'espai de treball.
Font: Elaboració pròpia.

Pel que respecta als recursos corresponents amb material i necessitats d'oficina, s'amortitza (vida útil) en vint anys i, en el cas del transport, en catorze anys [20].

Recursos indirectes	Cost	Ús per any	Dedicació al dia	Ús en el projecte	Amortització [€]
MICKE (escriptori [21])	69€	220	7	1210.75	2.71€
RENBERGET (cadira [22])	49€	220	7	1210.75	1.93€
T-jove (transport [23])	80€ / 3 mesos	220	7	1210.75	251.58€
Electricitat	0.24843€ / kWh [24]	220	7	1210.75	253.13€
Internet	53 € / mes	220	8	1210.75	437.52€
TOTAL					946.87€

Taula 7: Mostra l'amortització de cada recurs indirecte.
Font: Elaboració pròpia

Per realitzar el càlcul del cost de l'electricitat, s'ha de tenir en compte que la potència del portàtil és de 150W i s'amortitza a vint anys, per tant, el seu cost a vint anys és equivalent a: 0.24843€ / kWh × 0.15kW × 24h / dia × 30 dies / mes × 12 mesos / any × 20 anys = 6439.31€.
En el cas de l'internet a vint anys seria: 53€/mes × 12 mesos / any × 20 anys = 12720€.
I per últim, pel que respecta al transport, com s'amortitza a catorze anys, s'obté un cost de: 80€ / 3 mesos × 12 mesos / any × 14 anys = 4480€.

Per tant, si es fa la suma total del cost dels recursos indirectes, s'obté 282.68€ (cost de l'espai de treball) + 946.87€ (amortització de necessitats i material d'oficina, així com el transport) = 1229.55€.

6.1.4. Contingència

Per a les contingències, s'establirà un percentatge del 15%. Per tant, els costos de cada tipus de recurs s'obté de la següent taula:

Recurs	Cost [€]	Contingència [%]	Cost final [€]
Recursos humans	14846.77€	15	17073.79€
Recursos materials	445.84€		512.72€
Recursos indirectes	1229.55€		1413.98€
TOTAL			19000.49€

Taula 8: Mostra els costos aplicant les contingències.
Font: Elaboració pròpia

6.2. Estimació dels costos inicial

Pel que respecta als riscos, ja s'han tingut en compte a l'apartat 5.4. Gestió de risc: Plans alternatius i obstacles, sobreestimant algunes tasques. Tot i això, existeix la possibilitat (baixa) d'avaria del portàtil amb un cost estimat de 100€.
A més, com es va esmentar, existeix el risc de la disponibilitat dels servidors, per tant, també es tindrà en consideració afegir 2 servidors (on el cost és 2 servidors × 22 € / mes × 4 mesos=176€). També existeix l'opció de, en comptes d'afegir més servidors, agafar servidors més potents, i el cost d'ambdues solucions és molt similar (probablement, aquesta és la millor solució mediambientalment parlant). Per tant, es farà una estimació contemplant aquests riscos. Vegeu la taula 8:

Imprevist	Cost estimat [€]	Probabilitat [%]	Cost [€]
Reparació portàtil	100€	5	5€
Disponibilitat servidors	176€	30	52.8€
TOTAL			57.8€

Taula 9: Mostra els costos dels possibles imprevistos.
Font: Elaboració pròpia

Per tant, un cop presentats tots els costos del projecte aplicant les respectives contingències, en la taula 10 es mostra el cost total del projecte, que correspon a 19058.29 €.

Tipus de cost	Cost [€]
Recursos humans	17073.79€
Recursos materials	512.72€
Recursos indirectes	1413.98€
Imprevistos	57.8€
TOTAL	19058.29€

Taula 10: Mostra el pressupost del projecte.
Font: Elaboració pròpia

6.3. Estimació dels costos final

Un cop acabat el treball, l'estimació dels costos ha estat molt acurada, ja que, l'únic que ha canviat, és que, a l'últim mes de desenvolupament, es va haver d'incrementar la potència dels servidors a la següent instància, passant a valer de 22 € / mes a 42 € / mes. Fet que provoca que, els costos dels servidors passin a valer:
3 servidors × 22 € / mes × 3 mesos =198€;
3 servidors × 42 € / mes × 1 mes = 126€;
Obtenint llavors un cost de 324€, tenint una diferència al cost previ de 324€ - 264€ = 60€. Com es pot observar, aquest creixement de 60€ ja es va tenir en compte per si sorgia algun imprevist com aquest, exactament de 57.8€.
Tot i això, es farà el nou càlcul d'amortització i el cost dels recursos materials, per obtenir el pressupost final:
Amortització dels servidors: 36.68€.
Nou cost dels recursos materials: 378.27€ + 82.55€ = 460.82€. Aplicant les contingències (s'apliquen igualment, ja que el cost de S3 és aproximat), obtenim: 529.94€.

Per tant, el nou i definitiu pressupost final (sense comptabilitzar els imprevistos) és: 17073.79€ + 529.94€ + 1413.98€ = <u>19016.71€</u>. Pressupost molt semblant a l'estimat inicialment.

6.4. Control de gestió

En aquest apartat es mostren els descriptors numèrics que s'havien generat per tal de tenir controlades les possibles desviacions que es podien patir. Però, com s'ha explicat en la Planificació final, el temps emprat per cada tasca és molt semblant al planificat, per tant, els resultats de cada un d'aquests descriptors es posaran com més o menys (≈) igual a 0. Tot i això, com bé s'ha explicat a l'Estimació dels costos final, el cost dels recursos materials ha estat incrementat, per tant, s'obté:

- Desviació recursos humans per tasca:
  $(cost_{estimat} - cost_{real}) \times hores_{reals} \approx 0$ €

- Desviació en la realització de les tasques:
  $(hores_{estimades} - hores_{reals}) \times cost_{real} \approx 0$ €

- Desviació total dels costos dels recursos humans:
  $Cost\ RH_{estimat} - Cost\ RH_{real} \approx 0$ €

- Desviació total dels costos dels recursos materials:
  $Cost\ material_{estimat} - Cost\ material_{real} = 264$ € $- 324$ € $= -60$ €.

- Desviació total dels costos dels recursos indirectes:
  $Cost\ indirecte_{estimat} - Cost\ indirecte_{real} \approx 0$ €

- Desviació total dels imprevistos:
  $Cost\ imprevistos_{estimat} - Cost\ imprevistos_{real} = 57.8$ € $- 60$ € $= -2.2$ €

- Desviació en el total d'hores:
  $Hores_{estimades} - Hores_{reals} \approx 0$ hores

- Desviació del pressupost del projecte:
  $Cost\ total_{estimat} - Cost\ total_{real} = 19058.29$ € $- 19016.71$ € $= 41.58$ €

Com es pot observar, les desviacions patides en els costos dels recursos materials han estat bastant bé suplides pel cost dels imprevistos, per això, la desviació del pressupost del projecte és molt petita.

7. Informe de sostenibilitat

En tot projecte cal fer una anàlisi de la seva sostenibilitat tenint en compte tres dimensions diferents: econòmica, ambiental i social. Per tant, en aquest apartat es farà una autoavaluació sobre competències en sostenibilitat i es respondran unes preguntes sobre aquestes tres dimensions esmentades.

7.1. Autoavaluació

Gràcies a l'enquesta realitzada sobre sostenibilitat, puc extreure algunes conclusions sobre els meus coneixements i competències en aquest àmbit.

M'hi he adonat que existeixen moltes mètriques que no sóc capaç de poder analitzar, tot treballant en aquest àmbit professional. En el que més incapacitat tenia era en els indicadors relacionats amb el medi ambient. Tinc molt present l'impacte de tenir molts servidors per poder fer funcionar una web i les seves característiques, però no tenia present molts indicadors que m'hi he trobat a l'enquesta.

Respecte a l'aspecte social, el fet de treballar en el desenvolupament de software, donant millors ordenacions en les cerques dels usuaris, entre d'altres, provoca que el fet d'aportar noves eines i funcionalitats a la societat generi un progrés en aquest àmbit.

A més, amb el temps invertit a generar un pressupost ben detallat i realitzar l'enquesta i veure els indicadors que desconeixia, he aconseguit millorar l'aspecte econòmic.

7.2. Dimensió ambiental

Referent a PPP: Has quantificat l'impacte ambiental de la realització del projecte? Quines mesures has pres per reduir l'impacte? Has quantificat aquesta reducció?

Com bé s'ha explicat a l'hora de fer el pressupost, s'han tingut en compte el cost de 3 servidors. L'electricitat d'aquests no s'ha tingut en compte, ja que no s'hi ha hagut de comptabilitzar per tal de fer el pressupost perquè va a càrrec de l'empresa de servidors OVHcloud (tot i que, segurament, part dels costos elèctrics ja es troben aplicats en el cost mensual del servidor).

Per tant, per tal de quantificar l'impacte ambiental en kWh i CO_2 durant el projecte, es tindrà només en compte els recursos utilitzats per desenvolupar el projecte, exactament l'ordinador portàtil i els 3 servidors utilitzats pel cercador.

Pel cas dels servidors, tenim que emet 200 kg de CO_2 per servidor a l'any[25]. Com se sap que han estat utilitzats durant quatre mesos i es tenen 3 servidors, s'obté:
3 servidors ✕ 200kg CO_2 / servidor i any ✕ 1 any / 12 mesos ✕ 4 mesos = 200 kg CO_2 = 0,2 t CO_2.
En el cas del consum energètic, no hi ha dades oficials per part de OVHcloud, per tant, se suposarà que es consumeixen 1800 kWh a l'any per servidor[26]. S'obté, llavors:
1800 kWh / servidor i any ✕ 1 any / 12 mesos ✕ 4 mesos ✕ 3 servidors = 1800 kWh.
Per altra banda, es té l'ordinador, el qual consumeix 150 W i ha estat operatiu 7 hores a la setmana durant 6 mesos ≈ 910 hores. Per tant, s'obté un consum de:
150 W ✕ 910 h = 136500 Wh = 136,5 kWh.
I, per tal d'obtenir el consum energètic, s'assumeix que 0,25 kg CO_2/kWh[27]:
0,25 kg CO_2/kWh ✕ 136,5 kWh = 34,125 kg CO_2 = 0,034 t CO_2.

Llavors, si s'ajunta tot, s'obté un consum energètic de 136,5 kWh + 1800 kWh = <u>1936,5 kWh</u> i unes emissions de CO_2 de 200 kg CO_2 + 34,125 kg CO_2 = <u>234,125 kg CO_2</u>.

A mesura que hi havia més tràfic en els servidors degut a les interaccions dels usuaris i transformacions de les dades del cercador, es necessitava més potència per tal de poder aguantar l'augment de peticions. Per tant, la mesura que es va prendre per reduir l'impacte ambiental va ser incrementar la potència d'aquests servidors en comptes d'agafar més. Se sap que afegir un altre servidor hauria suposat un increment d'emissió de CO_2 de 200 kg CO_2 / any ✕ 1 any / 12 mesos ✕ 4 mesos = 66,67 kg de CO_2 i un consum energètic de 1800 kWh / servidor i any ✕ 1 any / 12 mesos ✕ 4 mesos = 600 kWh. Per tant, incrementant la potència d'aquests servidors, s'hauran incrementat una mica les emissions de CO_2 i consum energètic (no existeixen xifres oficials quantificables), però no tant com el que s'ha estalviat si s'hagués afegit un de nou.

Referent a PPP: Si fessis de nou el projecte, podries fer-ho amb menys recursos?

Si s'hagués de fer de nou el projecte, aquest no es podria haver fet amb menys recursos, ja que l'ordinador portàtil utilitzat pel desenvolupament d'aquest era necessari, i, els servidors, també. A més, a mesura que s'anava necessitant més potència en els servidors, aquesta s'anava incrementant, reduint així des d'un inici les emissions de CO_2 i consum energètic.

Referent a la Vida Útil: Quins recursos estimes que s'usaran durant la vida útil del projecte? Quin serà l'impacte ambiental d'aquests recursos?

Els recursos que hauran de continuar usant-se per tal que el cercador pugui seguir funcionant són els tres servidors. Si en un futur es volgués millorar el cercador, també s'hauria de tenir en compte l'ordinador portàtil (o un altre ordinador), però no es tindrà en compte en aquesta quantificació.
Per tant, se suposarà que la vida útil que tindrà el projecte és de set anys. Llavors, s'obté que l'emissió de CO_2 serà de:
200 kg CO_2 / 4 mesos ✕ 12 mesos / 1 any ✕ 7 anys = 4200 kg de CO_2 = 4,2 t de CO_2.
I el consum energètic de:
1800 kWh / servidor i any ✕ 7 anys ✕ 3 servidors = 37800 kWh = 37,8 MWh.

Referent a la Vida Útil: El projecte permetrà reduir l'ús d'altres recursos? ¿Globalment, l'ús del projecte millorarà o empitjorarà la petjada ecològica?

El projecte utilitza tres servidors, els quals ja estaven sent utilitzats a Tiendeo. Tot i això, al haver de necessitar més potència que l'anterior, empitjorarà, una mica, la petjada ecològica.
En un futur, el projecte podria reduir el nombre de servidors (recursos), només en el cas que les dades que es pugen als servidors d'Elasticsearch i/o les navegacions dels usuaris disminueixin.

Referent als Riscos: Podrien produir-se escenaris que fessin augmentar la petjada ecològica del projecte?

Hi ha dos casos en els quals puguin incrementar la petjada ecològica.
Un, si els servidors no poguessin amb la seva actual potència, s'hauria d'incrementar aquesta, fent així incrementar, una mica, la petjada ecològica.
L'altre, que no hi hagués forma d'incrementar més la potència dels servidors (ja que pot arribar un punt que el hardware no sigui prou) i, per tant, s'hagi d'incrementar el nombre de servidors. Aquest és un cas el qual és poc probable, però l'increment de la petjada seria considerable en comparació amb el cas anterior (exactament, com s'ha calculat prèviament, cada servidor a afegir incrementaria el consum energètic en 600 kWh i l'emissió de CO_2 en 66,67 kg).

Les raons dels casos previs podrien ser que el nombre de dades que se'n pugen als servidors i/o els usuaris de Tiendeo creixen enormement.

7.3. Dimensió econòmica

Referent a PPP: Has quantificat el cost (recursos humans i materials) de la realització del projecte? Quines decisions has pres per reduir el cost? Has quantificat aquest estalvi?

Els recursos materials són 529.94€ (com s'ha explicat a Estimació dels costos final) i els recursos humans són 17073.79€.

Les decisions que s'havien pres per reduir el cost ha estat una, la qual era intentar aguantar al màxim amb el nivell de potència que tenien els servidors, fins que, a l'últim mes, es va haver de pujar la potència per problemes tècnics. Per tant, el cost estalviat en aquests tres mesos a estat de 20 € / mes i servidor. Obtenint un estalvi de:
20 € / mes i servidor × 3 servidors × 3 mesos = 180 €.

Referent a PPP: ¿S'ha ajustat el cost previst al cost final? Has justificat les diferències (lliçons apreses)?

El cost previst al cost final ha estat pràcticament idèntic (com s'ha comentat a l'apartat Estimació dels costos final).
El fet d'afegir al pressupost inicial possibles imprevistos, ha fet que el pressupost final sigui pràcticament igual al previst. Fet que emmarca més la importància d'incloure possibles imprevistos en un pressupost de projectes grans, i més si aquests imprevistos poden ser majors.

Referent a la Vida Útil: Quin cost estimes que tindrà el projecte durant la seva vida útil? Es podria reduir aquest cost per fer-lo més viable?

Si se suposa que la vida útil del projecte és de set anys, el seu cost serà el dels servidors funcionant en aquests set anys, el qual correspon a:
42 € / mes i servidor × 12 mesos / 1 any × 7 anys × 3 servidors = 10.584€.
Aquest només podria ser reduït en el cas que les navegacions dels usuaris i/o les dades que se'n pugen als servidors baixessin considerablement, fets que són poc probables.

Referent a la Vida Útil: S'ha tingut en compte el cost dels ajustos / actualitzacions / reparacions durant la vida útil del projecte?

S'ha tingut en compte durant el desenvolupament del projecte, ja que va relacionat amb els recursos humans, més exactament amb el rol de System Engineer, ja que era l'encarregat del procés per millorar les instàncies dels servidors i problemes tècnics que podien sorgir.
Tot i això, si es volgués fer una aproximació del seu cost en el projecte, suposant que ha d'intervenir una hora al mes durant set anys, s'obté un total de:
1 hora / mes × 12 mesos / 1 any × 7 anys = 84 hores. Per tant, com es té l'aproximació del sou en hores, s'obté que el cost de possibles ajustos seria de:
84 hores × 23.43€ / 1 hora = 1968.12€.

Referent als Riscos: Podrien produir-se escenaris que perjudiquen la viabilitat del projecte?

L'únic risc que pot posar en dubte la viabilitat del projecte és que, segons els usuaris, no es facin bé les ordenacions. Actualment, com es veurà a l'apartat Validació dels resultats, els resultats són bons i fiables, tot i això, podria donar-se el cas que les ordenacions deixessin de ser les correctes, sigui per descriure malament, en format text, el que representa un article i/o sale (aquesta representació es veurà amb més detall a l'apartat Camps que defineixen un document) o per altres raons externes.

7.4. Dimensió social

Referent a PPP: La realització d'aquest projecte ha implicat reflexions significatives a nivell personal, professional o ètic de les persones que han intervingut?

El projecte ha ajudat a l'autor a introduir-se en l'àmbit professional com a Data Engineer. Ha après tecnologies que molt segurament haurà de continuar utilitzant en futurs treballs tals com Jenkins, Kubernetes, Python (més en profunditat) i, si es donés el cas, també Elasticsearch. Gràcies a aquest, s'han hagut de cercar solucions sense saber si eren les correctes o no i sense que cap persona ho confirmés.

A més, en haver d'interactuar amb diferents equips a l'empresa, s'ha conegut a gent molt ben formada i, independentment de l'edat, amb ganes d'aprendre i sense por a preguntar si hi ha alguna cosa que no entén. Aquest ambient de treball ha ajudat a créixer en l'àmbit personal i ètic, tractant de posar les mateixes ganes que la gent que ha participat en el projecte (o sense que necessàriament participi).

El gran repte ha estat poder entendre tota l'arquitectura del projecte en la qual s'anava a incorporar el cercador, la posterior selecció i transformació de les dades amb les quals anava a treballar el cercador i les decisions preses per generar la query de cerca que millor s'adaptava als camps seleccionats a partir de les dades.

Referent a la Vida Útil: Qui es beneficiarà de l'ús del projecte? Hi ha algun col·lectiu que es pot veure perjudicat pel projecte? En quina mesura?

El beneficiari principal del projecte és Tiendeo, ja que, a millor vagi el seu cercador, més clics en la seva web tindrà, i, per tant, els clients i publicistes de la web poden pagar més i/o mantenir-se en la web per la seva fiabilitat.
En ser un cercador amb l'objectiu d'oferir a la gent el que busca, no hi ha cap col·lectiu que es pugui veure perjudicat arrel al projecte.

Referent a la Vida Útil: En quina mesura soluciona el projecte el problema plantejat inicialment?

El problema que es tenia inicialment era que, com últimament els articles i sales se n'anaven taguejant malament, sigui per errors humans o pels bots que els generaven de forma automàtica, els resultats de les ordenacions anaven empitjorant. Gràcies al cercador, ara s'ofereixen tots aquests articles i sales que, amb el cercador antic no es mostraven, mostrant així més resultats i, per tant, millorant l'ordenació. A més, a part de solucionar el problema, s'ha oferit una nova possibilitat als usuaris, permetent poder cercar per qualsevol mena de text.
Llavors, es podria dir que el projecte soluciona completament el problema inicialment plantejat.

Referent als Riscos: Podrien produir-se escenaris que fessin que el projecte fos perjudicial per a algun segment particular de la població?

No hi ha cap cas possible que fos perjudicial per als usuaris, ja que al cap i a la fi, s'estan oferint articles. Podria passar que algun article fos violent/ofensiu arran de la generació automàtica d'aquests i, per tant, es procediria a l'eliminació instantània d'aquest.

Referent als Riscos: Podria crear el projecte algun tipus de dependència que deixés als usuaris en posició de debilitat?

La dependència que té el projecte és que els servidors que ofereixen els resultats als usuaris deixés de funcionar temporalment (la cerca per qualsevol mena de text, ja que les paraules precalculades es troben en la memòria del servidor de la web). Per intentar que no passi això, diàriament s'intenta revisar el funcionament dels servidors i, si es veiés que la potència d'aquests no és suficient, es tractaria d'incrementar la potència.

8. Anàlisi de diferents tecnologies

En aquest apartat es justificarà l'elecció de les diferents tecnologies tenint en compte les condicions que l'empresa Tiendeo té. Es veurà més a fons la raó d'escollir utilitzar Elasticsearch com el Search Engine que s'adequa més al projecte proposat per Tiendeo (en comptes d'altres solucions disponibles), traient el fet que Elasticsearch ja està implementat a l'empresa com una tecnologia en ús, però clarament no està sent utilitzada com a cercador.
Per últim, també se sospesarà si és necessari fer un canvi en les pipelines administrades per Kubernetes, tot sabent que és un sistema molt desenvolupat i la millora hauria de ser molt gran.

8.1. Search engines al mercat

Com bé s'ha explicat prèviament, existeix la necessitat d'utilitzar un Search Engine per a poder desenvolupar el cercador, ja que aquests, fa més de cinc anys que es troben en desenvolupament per tal de poder fer cerques entre milions de documents en qüestió de mil·lisegons.
Per tant, ara es veurà quines són les tecnologies més punteres del mercat. Com s'ha vist a la figura 4, en aquesta, es trunca fins al top 10 de les tecnologies més utilitzades i populars de les centes que es troben actualment.
Fixant-se bé, es veu que les quatre primeres s'han mantingut fixes durant un any sencer i el Top 3 està altament diferenciat en popularitat de la resta.

Per tant, s'analitzaran els tres primers search engines més utilitzats i amb més suport actualment. Es veuran característiques de cada un com el preu, la data de sortida a mercat (ja que pot ser indicador de maduresa del producte), manteniment, escalabilitat, llenguatges de programació que suporta, la documentació que té, com es realitza la implementació i característiques extres d'aquests.

8.1.1. Elasticsearch

El primer de tots que es mirarà serà Elasticsearch, fixant-se en el preu, primer es veu que Elasticsearch és Open Source, per tant, és gratis l'ús de la llicència. Si es volgués utilitzar en un servei gestionat cloud públic (tals com AWS, Google Cloud...), costaria entre 16 i 30 dòlars al mes, però com Tiendeo ho té muntat no cal aquest servei.
Elasticsearch va ser llançat l'any 2010 i es troba en constant manteniment amb diferents versions, sent l'última i actual la 7.16.
Quant a l'escalabilitat, és gran, ja que permet afegir nodes (servidors) a clústers (conjunt de nodes en els quals estaran distribuïts tots els documents) per

incrementar la capacitat i automàticament distribueix la càrrega de dades i quèries entre tots els nodes disponibles.
Pel que representa la programació, suporta els principals i més utilitzats llenguatges de programació, tals com: Python, Java, JavaScript, .Net… i és de fàcil implementació, ja que es basa en crides HTTP API REST. A més, la seva documentació és àmplia i extensa, com la comunitat que té darrere.

8.1.2. Splunk

Seguidament, es té Splunk, un cercador que ha anat creixent en els últims anys i ara està molt ben posicionat. Aquest és gratis si el volum d'indexaments diaris (nombre de documents que se'n pugen al Search Engine) és inferior a 500 MB. En el cas que s'indexin més dades que 500 MB, si el valor és més elevat, menys costarà, com veiem a la figura 7.

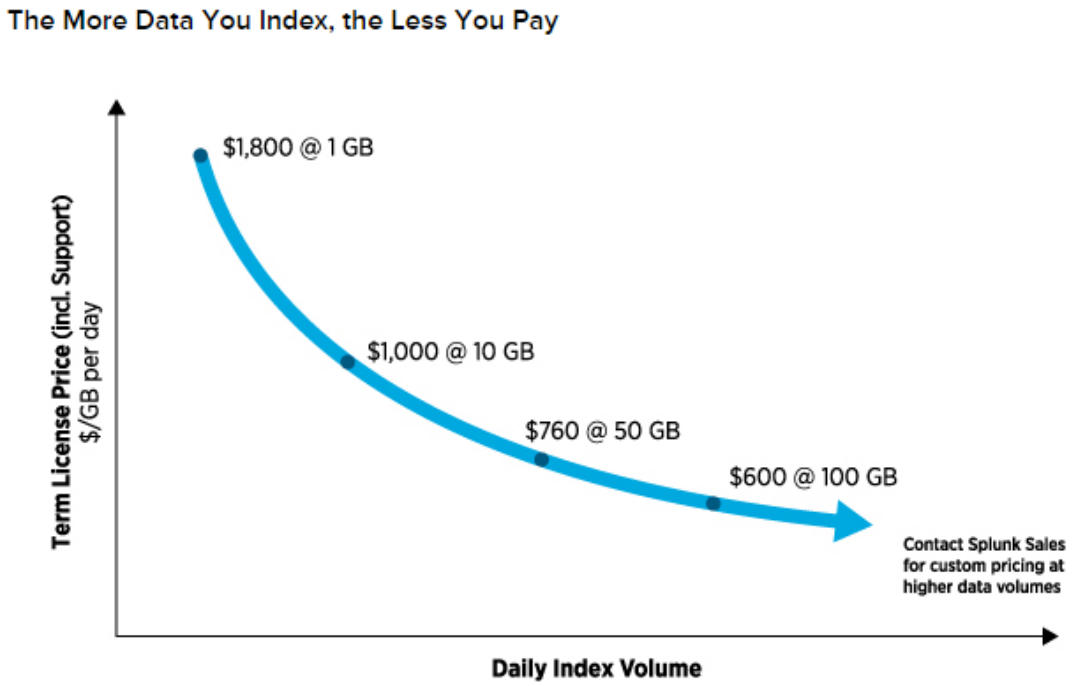


Figura 7: Mostra el preu pel volum de documents que s'indexen al cercador.
Font: https://www.learnsplunk.com/splunk-pricing---splunk-licensing-model.html

Al saber que aproximadament la quantitat de dades diàries que es volen emmagatzemar per tal d'anar actualitzant es troba al voltant de quatre a cinc gigabytes, automàticament aquest Search Engine queda descartat, ja que està pensat per organitzacions que indexen una quantitat gegant de dades diàriament (o molt poques dades).

8.1.3. Solr

Per últim es té Solr, un cercador el qual es troba entre els més usats i populars del mercat. Igual que Elasticsearch, Solr és Open Source i, per tant, és gratis. Encara que també té la versió Cloud, la qual va entre 9 i 54 euros al mes.
La seva data de sortida va ser l'any 2006, i, com també els anteriors, Solr està en constant manteniment i té moltes versions disponibles. La versió més actual és la 8.11.0.
Ambdós Solr i Elasticsearch estan basats en la llibreria Lucene, per tant, l'escabilitat que tenen és bastant similar. Com també els llenguatges que admet, tot i que difereixen alguns, però són dels menys populars tals com Groovy, Erlang… A més, la forma d'implementació també és utilitzant APIs, encara que la comunitat no és tan gran respecte Elasticsearch.

8.1.4. Search Engine escollit

Com bé ja s'havia esmentat anteriorment, el Search Engine escollit és Elasticsearch, ja que el preu d'Splunk seria massa elevat pel cas d'ús del cercador a desenvolupar i Solr no ofereix característiques disruptives comparades amb Elasticsearch, ja que, de fet, s'assemblen bastant i té una comunitat més petita.

Per tant, no hi ha cap Search Engine que ofereixi millores substancials comparades a Elasticsearch i que es trobi en el més utilitzat en el mercat. Llavors, no hi ha necessitat de canviar de tecnologia que s'està fent servir a Tiendeo. Això també provoca que no sigui necessari canviar de llenguatge de programació (Python), ja que és el llenguatge de programació amb el qual es programen les pipelines a Tiendeo i és el més usat i útil pel que respecta el tractament de dades[32].

8.2. Administradors de contenidors

Pel que respecta el funcionament de les pipelines, aquestes han estat desenvolupades per l'equip de DATA de Tiendeo utilitzant Python com a llenguatge de programació i Kubernetes per administrar els contenidors de Docker que generen aquestes pipelines. Per tant, utilitzen la combinació de Kubernetes amb Docker com a tecnologia.
En principi no hi ha cap intenció per part de Tiendeo en canviar aquesta tecnologia, tot i que si hi hagués una gran millora en alguna alternativa a Docker o a Kubernetes es podria contemplar aquesta opció.

Fixant-se en els administradors de contenidors, hi ha dos clars competitors, Kubernetes i Docker Swarm[15]. Si es comparen aquests dos, el guanyador seria

[15] Documentació de Docker Swarm. Disponible a: https://docs.docker.com/engine/swarm/

Kubernetes, ja que en temas com escalabilitat, disponibilitat, balanceig de càrrega, actualitzacions, emmagatzematge, etc., Kubernetes és superior menys en temes de complexitat en la implementació[33]. Tot i això, al tenir l'equip de SYSTEMS especialitzat en tecnologies com Kubernetes, aquesta última característica no és important.
Llavors, per això no és necessari fer un canvi en la tecnologia utilitzada en l'administrador de contenidors.

Per altra banda, existeixen diverses alternatives pels contenidors, tals com AWS Fargate[16], Google Kubernetes Engine[17], Amazon ECS[18], etc., i, entre tots aquests, el més adient és Docker, ja que és el més recomanat per a developers en camps orientats a la creació de diferents ambients (environments) de treball, a més d'estar treballant amb micro serveis (com cridar a diferents APIs en les pipelines)[34]. Per tant, igual que amb Kubernetes, no cal canviar la tecnologia utilitzada pels contenidors, Docker.

[16] Documentació de AWS Fargate. Disponible a: https://aws.amazon.com/es/fargate/
[17] Documentació de Google Kubernetes Engine. Disponible a: https://cloud.google.com/kubernetes-engine/
[18] Documentació de Amazon ECS. Disponible a: https://aws.amazon.com/es/ecs/

9. Desenvolupament del cercador

En aquest apartat s'observarà l'estil d'arquitectura que té el projecte en el qual s'està implementant el cercador, es veurà una visió general d'aquest, és a dir, mitjançant pipelines, de quina forma s'executen els processos necessaris per a poder tenir el cercador operatiu amb les dades actualitzades.
A més, es veurà com se seleccionen les dades a utilitzar, així com la configuració per cada idioma del cercador. També s'aprofundirà en la query usada per realitzar la cerca en els documents indexats a Elasticsearch i com es fa el càlcul de les paraules precalculades (d'aquí en endavant anomenades Collections).
I, per últim, s'acabarà veient l'autocompletat que s'usarà al cercador.

9.1. Arquitectura del projecte

Ara es passarà a explicar el funcionament de les pipelines i com estan distribuïts els mòduls de Python, és a dir, el projecte en el qual el cercador s'haurà d'implementar.

9.1.1. Disseny de les pipelines

Una pipeline pot tenir diferents fases, amb quatre tipus de fases principals, com es veuen a continuació:

- **Extract**: Com bé diu el seu nom, aquest tipus de fase serveix per extreure dades d'un lloc en específic. Aquest lloc pot ser de S3, una query SQL, Elasticsearch o fins i tot serveis externs com APIs de Google, etc. En passar d'una fase de tipus extract, a la següent, les dades extretes es passen com a paràmetre.

- **Parametrize**: serveix per passar com a paràmetre algunes variables a diferents fases de la pipeline. Per exemple, si s'observa la figura 8, la fase 1 (parametrize) podria enviar a la fase 2 el cos d'una crida a una API d'Elasticsearch. Per tant, a la fase 2, només s'executa la crida a l'API sense afegir cap lògica. D'aquesta manera es manté la coherència entre les diferents fases i mètodes.

- **Transform**: En aquest tipus de fase s'apliquen transformacions a les dades extretes prèviament. Per tant, per a executar una fase de tipus transform, ha de venir una fase prèvia d'extract.

- **Load**: Generalment, és l'última fase de la pipeline i serveix per pujar les dades prèviament extretes i, opcionalment, transformades, a un lloc en específic, sigui S3, Elasticsearch, etc.

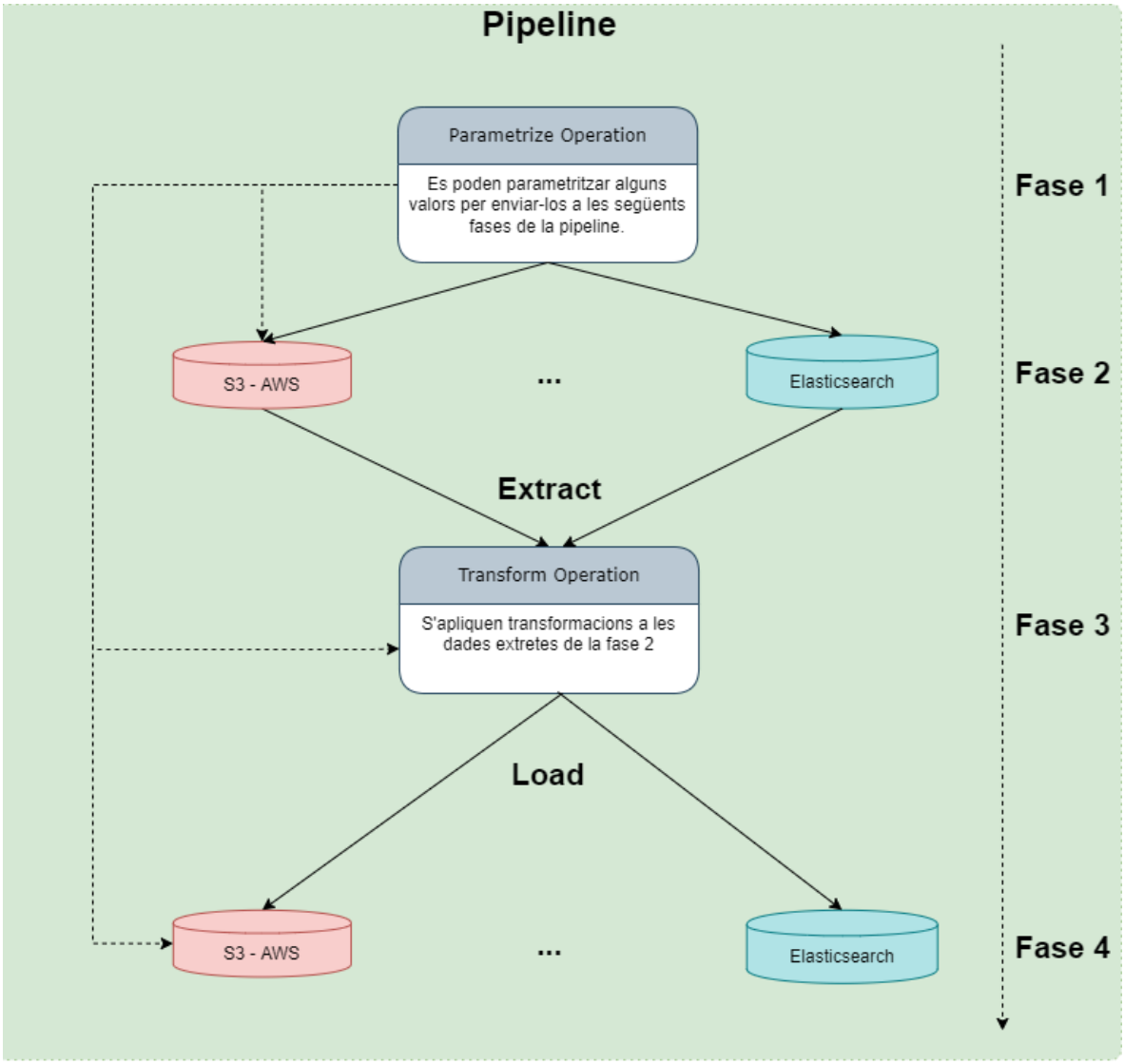


Figura 8: Mostra les fases les quals podria passar una pipeline.
Font: Elaboració pròpia.

Com es pot observar a la figura, un exemple de Pipeline pot ser com la que es mostra. Es parametritzen valors per passar com a paràmetre a les següents fases, s'extreuen dades, sigui d'S3, Elasticsearch o qualsevol altre mètode. Aquestes dades extretes es transformen i posteriorment se'n pugen a S3, i/o a Elasticsearch, i/o qualsevol altre lloc on es puguin emmagatzemar dades.

Totes les pipelines poden ser iterades per una funció que es pot definir. En el cas del Search Engine, també s'haurà d'iterar la mateixa pipeline per tots els països en els quals Tiendeo treballa. Vegeu la següent figura 9 d'un exemple real d'una pipeline en format YAML:

```yaml
group: SearchEngines
name: Article

country_iteration: true
transform_iteration:
  - method: operation
    group: SearchEngines
    name: Article

phases:

  - type: parametrize
    method: operation

  - type: extract
    method: s3
    bucket_name: tiendeo.data.profiling.pro
    key_prefix: Scores/Article/Current/country=${country}/modeltype=${model_type}/modelname=${model_name}/

  - type: transform
    method: operation

  - type: load
    method: elasticsearch
    index_alias: ${env_lower}_profiling__search_engines_article__${country_lower}
    find_index_by_alias: true
    index_type: _doc
    elastic_environment_name: profiling-search-engines
    index_document_mapping:
      method: scheme
    _add_inferred_column_types_to_base_column_types: False
```

Figura 9: Mostra la pipeline utilitzada per pujar les dades a Elasticsearch.
Font: Elaboració pròpia.

Com s'observa, en aquesta pipeline està activada la funció per iterar per tots els països (country_iteration) i està definida una altra funció d'iteració (transform_iteration). Per tant, s'executaria la pipeline tantes vegades com països × iteracions surtin del transform_iteration.
Com es pot interpretar, les fases són: parametrize, extract d'S3, transform, i, per últim, load a Elasticsearch.

9.1.2. Distribució de carpetes

Per a poder tenir una visió de l'arquitectura del projecte en el qual el cercador està implementat, vegeu la següent figura 10:

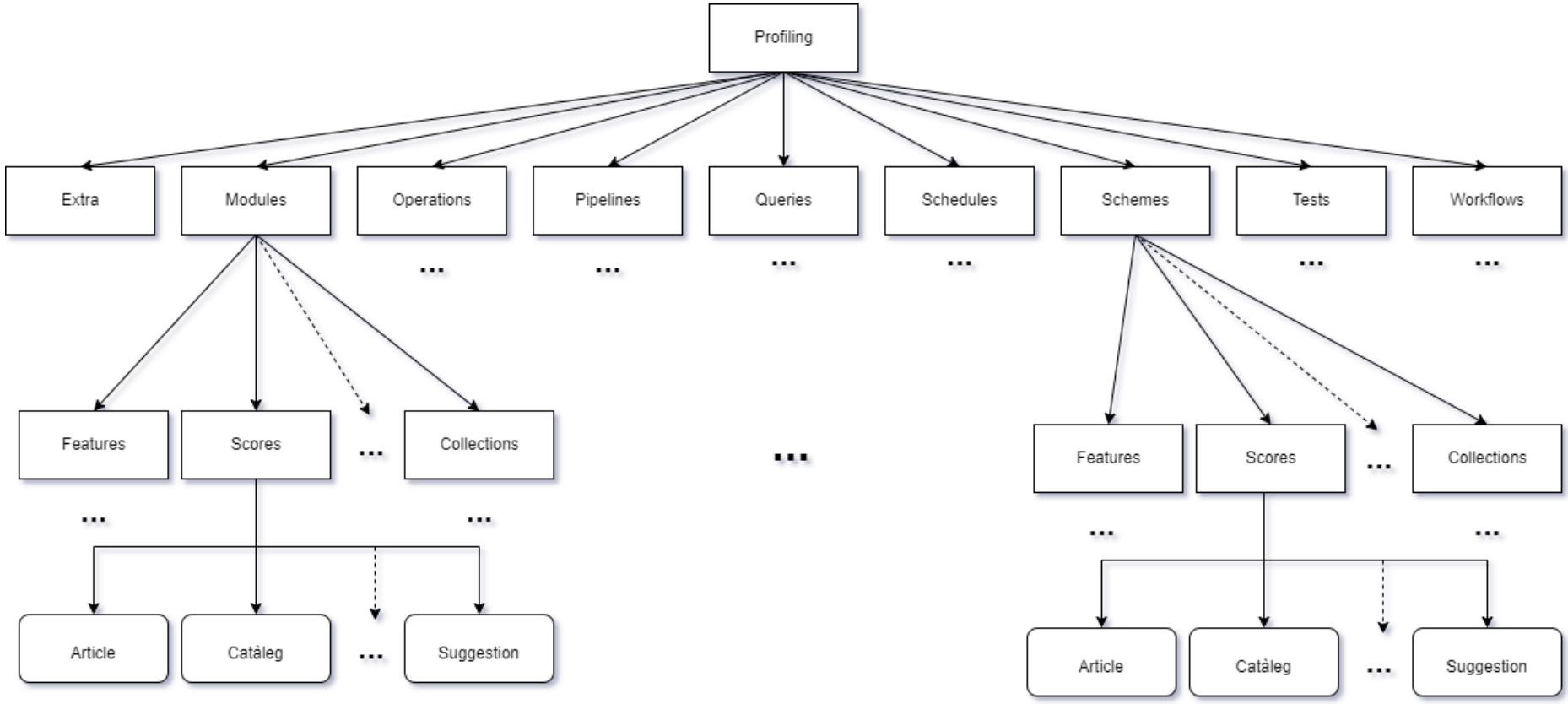


Figura 10: Mostra l'estructura de les carpetes del projecte.
Font: Elaboració pròpia.

Com es pot observar, de la carpeta del projecte, anomenat Profiling, pengen diferents subdirectoris, els quals s'expliquen a continuació:

- **Extra**: En aquesta carpeta es troben fitxers que marquen les dependències requerides per a poder executar el projecte, requeriments de llibreries del mateix projecte i scripts d'automatització.

- **Modules**: Es troben mòduls de suport pels Endpoints de l'API del projecte.

- **Operations**: És la carpeta on es defineix totes les operacions de tipus Transform i Parametrize, és a dir, és on es programa el tipus de transformació de les dades que es vol fer i les variables que es volen passar com a paràmetre.

- **Pipelines**: Es defineixen les pipelines en format YAML.

- **Queries**: Es troben les queries que s'executen a Athena de tipus SQL, queries que s'executen a Elasticsearch de tipus YAML, etc.

- **Schedules**: Fitxers de tipus YAML els quals determinen quan i quantes vegades s'han d'executar les pipelines o altres processos.

- **Schemes**: Es troben fitxers de tipus YAML amb els camps de les dades que se'n pujaran a algun lloc. Per exemple, si es fa un load d'una taula a S3, s'ha de crear un fitxer de tipus YAML especificant els camps que tindrà la taula.

- **Tests**: Es troben scripts de Python per a testejar el correcte funcionament de les pipelines, els fitxers que creen aquestes, etc.

- **Workflows**: Es troben fitxers de tipus YAML de pipelines que executen per fases altres pipelines. És a dir, cada fase, de forma seqüencial, representa l'execució d'una altra pipeline. S'explica amb més detall i context al subapartat 9.2.7. Jobs automàtics de Kubernetes.

Menys a la carpeta d'Extra, totes les carpetes tenen la mateixa jerarquia dins. I, dins d'aquestes, els fitxers de les diferents entitats a les quals aplica (articles, catàlegs, etc.) i la seva operació corresponent. Es mostren les carpetes Features, Scores i Collections, que són les que s'expliquen en el següent subapartat, com també les entitats Article, Catàleg i Suggestion.
Uns exemples serien, trobant-se a Profiling/Pipelines/Collections/Article, hi haurà una pipeline que fa referència a l'entitat Article, amb les seves fases pròpies. Mentre que trobant-se a Profiling/Operations/Collections/Article, es troben funcions de Transform i Parametrize per generar la Collection i aquestes funcions es cridarien en la seva pipeline corresponent (Profiling/Pipelines/Collections/Article).

9.2. Visió general del cercador

En aquest subapartat, es faran representacions en format de diagrama i algunes definicions per tal de veure cada una de les fases per les quals el cercador passa.

Cada una de les pipelines que es mostren a continuació s'executen per cada país, ja que cada un té uns articles i sales diferents a mostrar.

9.2.1. Extracció de les Features

9.2.1.1. Definició

Per tal de donar més importància a uns articles i/o sales (d'aquí en endavant ítem, per tal d'englobar els dos junts) determinats, es calculen unes característiques (features) a partir d'estadístiques i contingut de Tiendeo. Pel cercador, es tenen les següents features (cada punt dels següents fan referència a un ítem):

- Si té descompte
- Si es troba en campanya
- Si està a prop d'expirar (ja que els ítems tenen una data d'inici i fi de publicació. Un cop passada la data de fi, l'ítem deixa d'estar públic)
- Si s'ha publicat recentment
- En el cas de ser un article, si aquest es troba a la portada del catàleg al qual fa referència
- Si l'ítem és una sale (ja que l'entitat sale és una nova entitat dins Tiendeo i es volen prioritzar una mica davant els articles)
- Si l'ítem té preu (com hi ha ítems que s'afegeixen mitjançant OCR[19], de, per exemple, catàlegs, pot passar que el seu preu no hagi estat detectat)
- El CTR[20] de l'ítem quan apareix com a resultat d'una cerca
- Si el retailer de l'ítem és important (és a dir, hi ha una llista ordenada dels retailers de més a menys importància)
- El percentatge de descompte de l'ítem, si és que té (descompte relatiu)
- El descompte, en euros (o la moneda del país), si l'ítem en té (descompte absolut)

La raó per la qual es té una feature del descompte relatiu i una altra del descompte absolut és perquè no és el mateix un descompte del 50% en un ítem d'1 € que un descompte del 50% en un ítem de 500 €, perquè els diners estalviats és clarament superior en el segon cas.
A més, per cada feature, s'ha de garantir que els valors es troben en el rang dels reals [0,1], ja que si no, aquestes no es poden comparar entre si. Llavors, aquestes s'hauran de normalitzar pel valor màxim entre tots els ítems de cada feature.

[19] Procés pel qual, s'identifica, de forma automàtica a partir d'una imatge, símbols i/o caràcters que pertanyen a un alfabet concret[35].
[20] Click-through rate. És el ràtio de clics per part dels usuaris en un ítem respecte a la quantitat d'impressions que ha tingut.

9.2.1.2. Mètode d'extracció

Per tal de veure la forma d'extracció d'aquestes features, es té la següent figura 11:

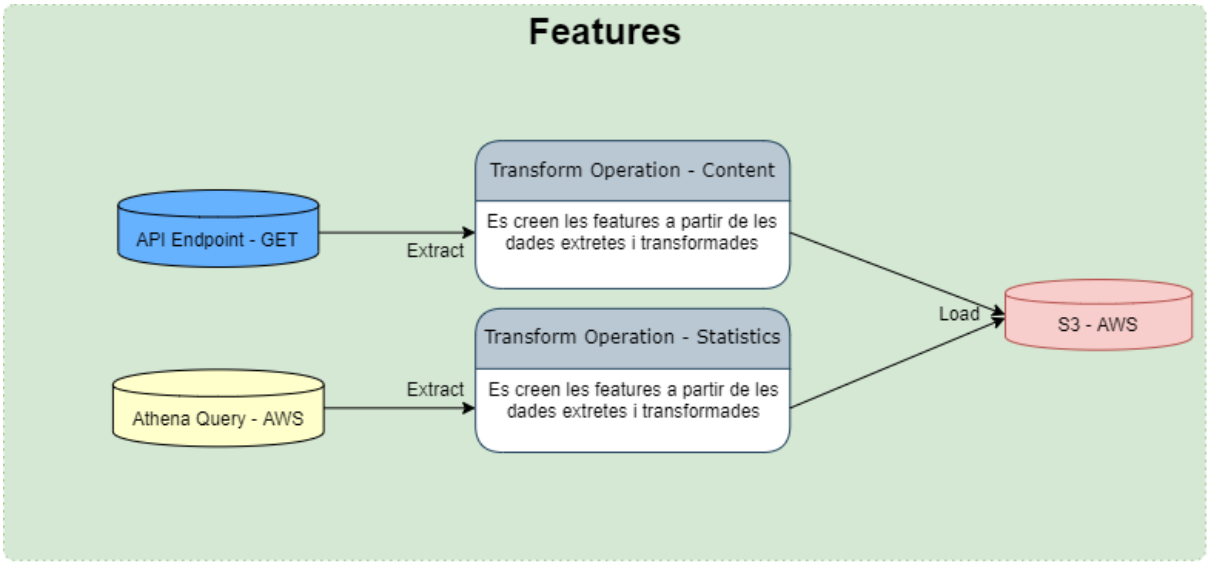


Figura 11: Mostra el procés per calcular les features.
Font: Elaboració pròpia.

La figura s'ha de veure com si fossin dues pipelines.
En una, es fa una crida a un Endpoint de l'API de l'equip de Content de Tiendeo. El que s'extreu d'aquí, són tots els ítems amb informació addicional necessària per calcular les features, per, després, a l'operació de Transform, generar totes les features per cada ítem. Un cop acabat aquest procés, se'n puja a S3 en un fitxer comprimit de tipus JSON per a poder ser utilitzat més endavant.
La segona pipeline, executa una query SQL agafant estadístiques de diferents taules de la BD de Tiendeo per cada ítem. Tot seguit, s'apliquen transformacions per generar les features per a cada ítem. Igual que abans, se'n puja a S3 en un fitxer comprimit de tipus JSON.
Com es pot observar, es fa una separació lògica entre les features de tipus estadística (que es treuen a partir d'estadístiques dels usuaris) i les de contingut (que es treuen de la BD de Tiendeo) per tal d'afegir un cert grau de congruència.

De forma resumida, si es tinguessin 200.000 ítems, 9 features de contingut i 2 features d'estadístiques, s'obtenen 200.000 ítems $\times$ (9 features$content$ + 2 features $statistics$) = 2.200.000 files ítem-feature.

9.2.2. Extracció dels Scores

9.2.2.1. Definició

Com s'ha explicat anteriorment, s'utilitzen unes features per a determinar la importància de cada ítem. Però, també es vol poder donar més importància a una

feature que a una altra. Per tant, per a fer això, en un Spreadsheet de Google, es posarà el nom de la feature, amb el seu pes i es normalitzarà per així obtenir un pes per feature entre els valors dels reals [0,1). Vegeu la següent taula 11:

Key	Status	RawValue	Value
content_has_discount	1	4	0.01568627
content_is_in_campaign	1	20	0.07843137
content_is_near_publication_end_date	1	1	0.00392157
content_has_been_recently_published	1	1	0.00392157
content_is_on_catalog_cover	1	2	0.00784314
content_product_score	1	3	0.01176471
content_brand_score	1	2	0.00784314
content_is_tag_box	1	15	0.05882353
content_is_sale	1	100	0.39215686
content_has_price	1	15	0.05882353
statistics_article_search_engine_ctr	1	30	0.11764706
statistics_article_clippings	1	14	0.05490196
content_brand_relative_discount_score	1	3	0.01176471
content_brand_absolute_discount_score	1	5	0.01960784
content_product_relative_discount_score	1	4	0.01568627
content_product_absolute_discount_score	1	6	0.02352941
content_is_top_retailer	1	30	0.11764706

Taula 11: Mostra el pes i el valor normalitzat de cada feature.
Font: Elaboració pròpia i Tiendeo.

Com es veu en la taula, cada feature, les quals han estat explicades prèviament, té un pes (RawValue) i el seu pes normalitzat (Value). La columna Status indica si la feature està activa (1) o no (0).
El càlcul del pes normalitzat és el valor RawValue$/\sum$RawValue.

Un cop llegits aquests valors, es computa, per cada ítem:
$\sum_{per\ tota\ feature}$ FeatureValue $\times$ Value. On, FeatureValue és el valor normalitzat de la feature i Value el seu pes normalitzat respecte les altres features. Per tant, s'obté un valor entre [0,1] el qual anomenem Score.

9.2.2.2. Mètode d'extracció

En aquest cas, per a poder generar els Scores per cada ítem, només cal una sola Pipeline, la qual extreu d'S3 l'arxiu generat anteriorment de les Features amb els seus valors i, del Spreadsheet, els pesos per cada feature. Un cop es tenen

aquestes dades, es calcula l'Score en l'operació de Transform i es genera un arxiu comprimit que se'n puja a S3 en format JSON. Vegeu la figura 12:

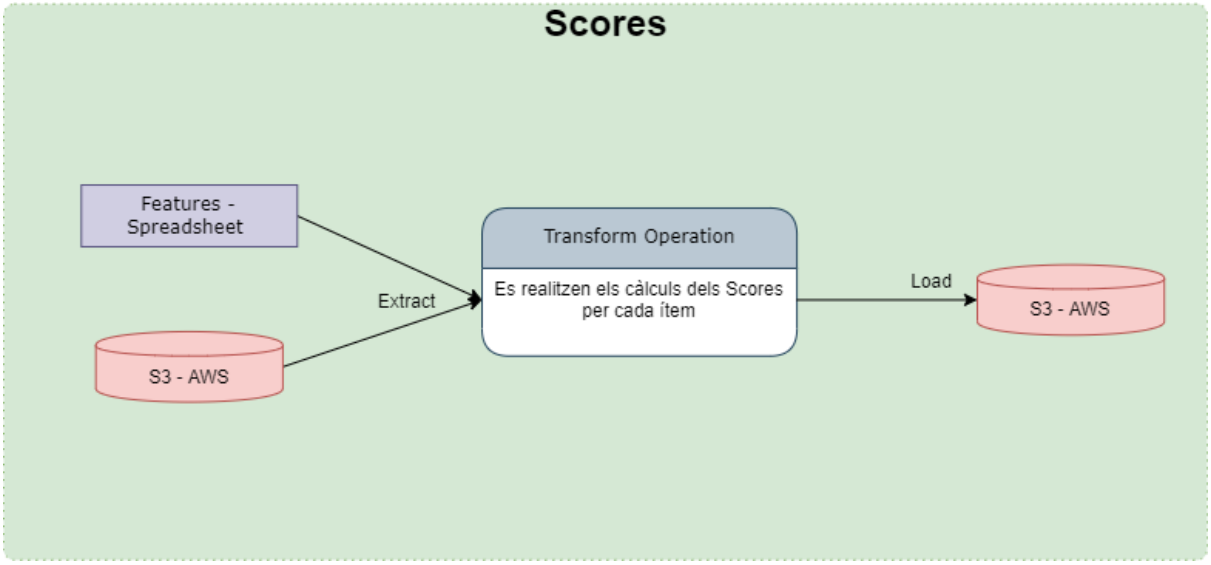


Figura 12: Mostra el procés per calcular els Scores.
Font: Elaboració pròpia.

Per tant, en aquest arxiu, es tindran tantes files com ítems hi hagi. I, a cada fila, es tindrà l'Score per l'ítem corresponent.

9.2.3. Pujada de les dades a ElasticSearch

Per tal de pujar les dades a Elasticsearch, s'extreuen els Scores prèviament calculats per la pipeline d'Scores. Amb aquest fitxer, es tenen tots els ítems que es poden oferir com a resultat.
Per tant, a la fase de Transform, s'afegiran els camps necessaris a aquests ítems (explicat amb més profunditat a l'apartat 9.3. Selecció de les dades a utilitzar) i es pujarà el clúster d'Elasticsearch com a índex.
En l'última fase, la de la pujada a Elasticsearch, es crea l'índex, els camps que aquest tindrà i els seus tipus i es realitza una crida a la Bulk API d'Elasticsearch per a poder indexar documents de la forma més ràpida possible. Vegeu la figura 13:

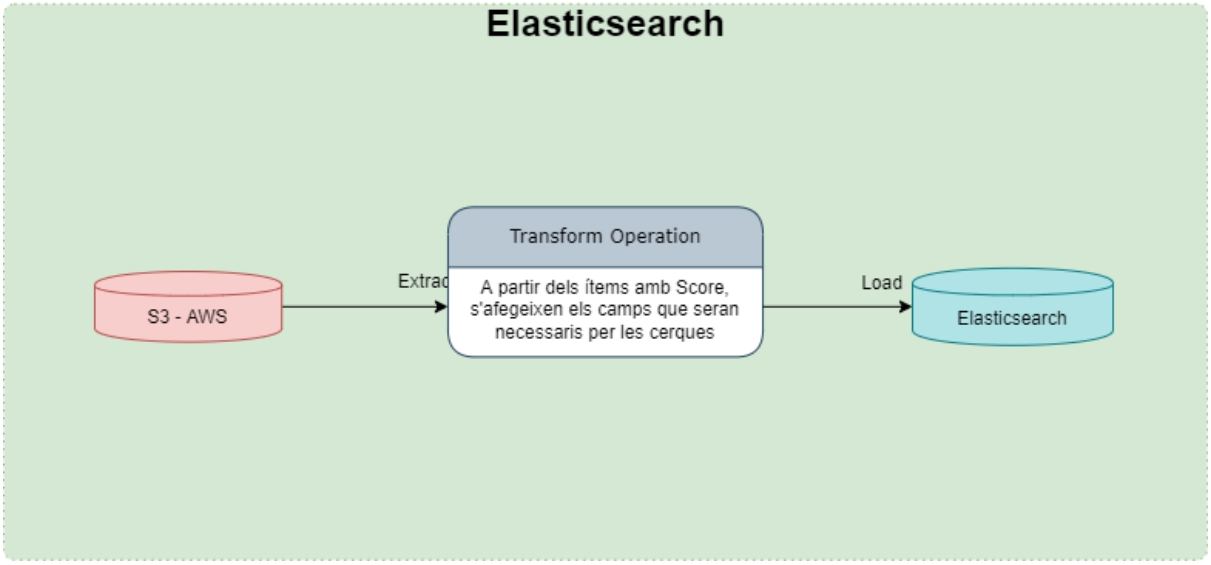


Figura 13: Mostra la pipeline de la pujada de les dades a Elasticsearch.
Font: Elaboració pròpia.

Vegeu que hi haurà un índex diferent per cada país, ja que cada un dels països té els seus propis ítems i configuracions d'Elasticsearch.

9.2.4. Creació de les Collections

Com s'ha comentat anteriorment, el cercador tindrà dues variants. Les cerques per paraules precalculades (Collections) i les cerques lliures (d'aquí en endavant cerques custom). Com s'explicarà al subapartat 9.2.6. Funcionament de l'autocompletat, s'incitarà a l'usuari a fer cerques les quals pertanyin a les Collections mitjançant l'autocompletat.

Les Collections són una llista de llistes. És a dir, per cada paraula que es vol precalcular, calcular les seves solucions enviant les cerques d'aquestes a Elasticsearch i guardar la seva resposta. Vegeu la següent figura 14:

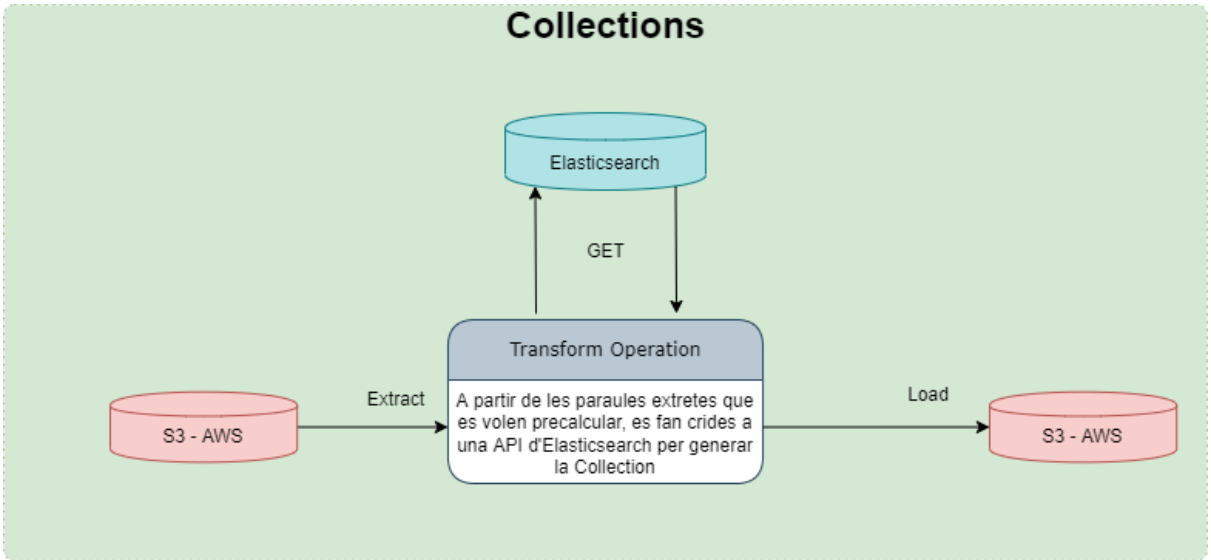


Figura 14: Mostra la pipeline per a generar les Collections.
Font: Elaboració pròpia.

Per entendre el diagrama anterior, primerament s'extreu de la BD de Tiendeo totes les paraules que facin falta precalcular pel cercador. En aquest cas, serien totes les marques i els productes existents.
Tot seguit, es prepara el cos d'una crida a la Multi search API[21] d'Elasticsearch, la qual permet paral·lelitzar l'execució de queries per retreure documents (exactament amb la query que es mostrarà a l'apartat 9.5. Creació de la query de cerca) i així poder generar les tuples paraula precalculada-llista de resultats de la forma més ràpida possible.
Un cop es neteja i es deixa en el format apropiat per a poder ser consumit, se'n puja el fitxer comprimit en format JSON a S3. També s'ha de tenir en compte que pel cercador hi ha dues Collections, per marques (Brands) i productes (Products). Per tant, aquesta mateixa pipeline s'executarà dues vegades per fer dos fitxers comprimits diferents: un per les marques i un altre pels productes.

9.2.5. Funcionament de l'autocompletat

L'objectiu de l'autocompletat és que si un usuari busca "Tel", li surtin suggerències de l'estil "Televisores", "Televisor Samsung", "Televisión", etc. Aquestes recomanacions que es mostren és una combinació de quatre entitats diferents: Retailers, Brands, Products i cerques custom. Vegeu la figura 15:

[21] Documentació Multi Search API. Disponible a:
https://www.elastic.co/guide/en/elasticsearch/reference/current/search-multi-search.html

Figura 15: Mostra l'actual autocompletat.
Font: Tiendeo.com

L'ordenació de l'actual autocompletat no segueix cap lògica (perquè el que fa és agafar la primera recomanació que es troba en una llista no ordenada). Per tant, el que es vol aconseguir és ordenar, donant més importància a algunes recomanacions i menys a unes altres (per exemple, si és una paraula precalculada, a aquestes se li dóna més importància). A més, afegir a aquesta llista les cerques custom dels usuaris que més buscades són (aplicant les transformacions necessàries per ajuntar les cerques més semblants mitjançant NLP i intentar eliminar possibles faltes ortogràfiques).

El funcionament per a generar aquesta llista és similar a l'utilitzat al Search Engine.

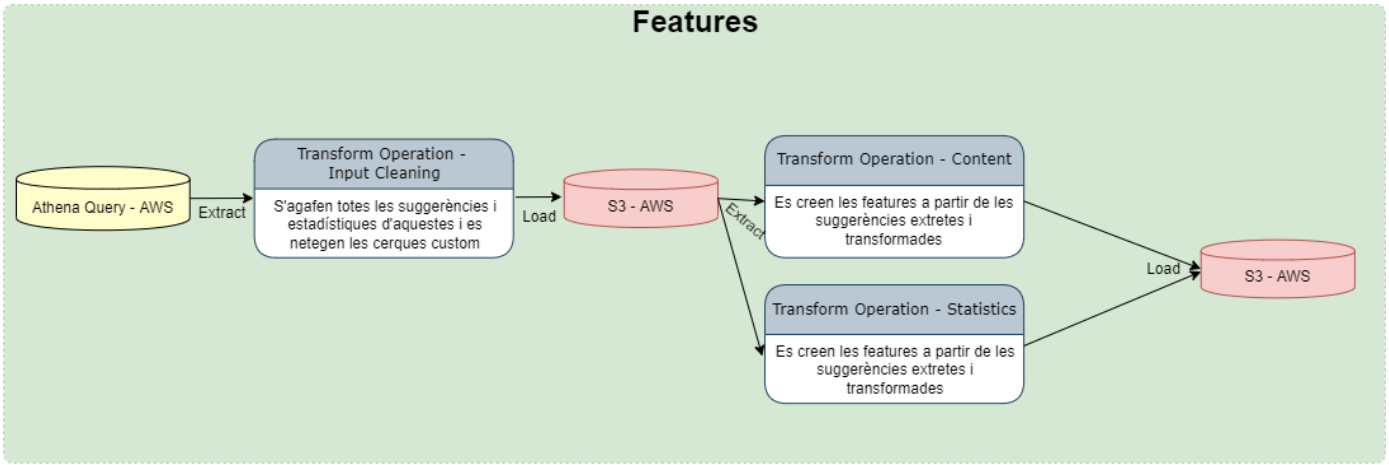


Figura 16: Mostra el càlcul de les suggerències i les features d'aquestes.
Font: Elaboració pròpia.

Com es veu a la figura 16, primerament, com les suggerències no es troben a cap lloc de Tiendeo, es realitza una query sobre les taules de la BD per extreure les paraules precalculades i les cerques custom dels usuaris, així com el nombre de vegades que les paraules i les cerques són buscades. Seguidament, es netegen i transformen les cerques dels usuaris i se'n puja un JSON comprimit a S3 per a tenir en la BD totes les suggerències existents.
Tot seguit, a partir de les suggerències pujades a S3, es calculen les features de tipus contingut i estadística i se'n puja a S3, igual que amb el Search Engine, tot i que ara en compte de ser una fila per ítem-feature, és suggerència-feature.

Les features aplicades són les següents (es troben normalitzades igual que abans):

- La popularitat de la suggerència (en quina mesura ha estat buscada, semblant al CTR)
- Popularitat de la suggerència separant per entitat. És a dir, és normalitzada separant entre productes, marques, retailers i cerques custom
- Si és un retailer important
- Si el retailer de la marca és important
- Si la categoria del producte és important (cada producte, com també les marques i retailers, té una categoria corresponent. Per exemple, un televisor correspon a la categoria "Informática y Electrónica", unes patates a "Supermercados", ...)

Segonament, es calculen els Scores. Igual que al Search Engine, s'assignen uns pesos a cada feature i es realitza, per cada una de les suggerències, el càlcul: $\sum_{per\ tota\ feature}$ FeatureValue $\times$ Value.
En aquest cas, el diagrama de com seria el disseny de la pipeline és igual al mostrat en el subapartat 9.2.2. Extracció dels Scores, tot i que en el transform es realitza el càlcul per les suggerències. Per tant, se'n puja un fitxer comprimit de tipus JSON a S3, on cada fila representa la tupla suggerència-score.
Com les últimes tres features no poden ser aplicades a les cerques custom, els scores de les suggerències per les cerques custom seran menors que per la resta d'entitats de les suggerències. D'aquesta forma, s'aconsegueix que la majoria de suggerències que són cerques dels usuaris tinguin un score menor, i, per tant, que surtin més a baix en la llista de l'autocompletat (aconseguint així donar més importància a les paraules precalculades).

En darrer lloc, tenim el càlcul de la Collection. En aquest cas només es té una sola Collection a generar, és a dir, en comptes de ser una llista de tuples paraula precalculada-llista de resultats, és una sola llista amb les suggerències ordenades decreixentment pel Score generat prèviament. Vegeu la figura 17:

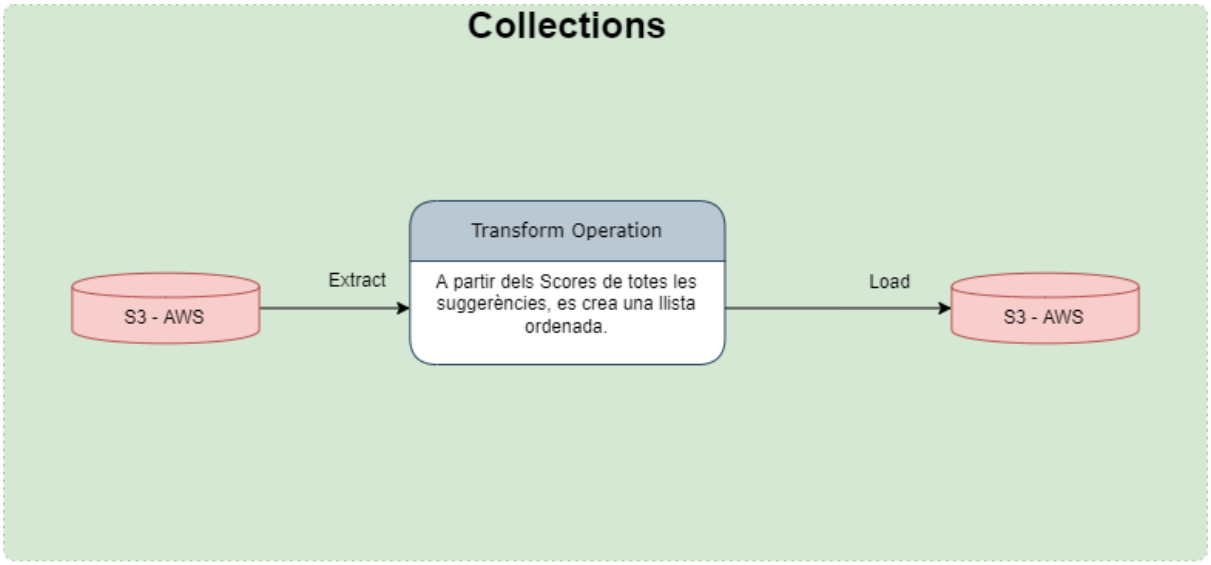


Figura 17: Mostra la pipeline per a calcular la Collection per l'autocompletat.
Font: Elaboració pròpia.

Primer, s'agafa d'S3 els Scores per cada suggerència. Un cop obtingut, al Transform es crea una llista ordenada decreixentment per l'Score. I, per últim, se'n puja en format JSON, aquest fitxer comprimit a S3.

9.2.6. Consumició del contingut

9.2.6.1. Search Engine

Per tal que l'equip de web passi a consumir la nova ordenació dels ítems pel cercador, s'utilitzen dos endpoints a l'API de Profiling de l'equip de data, un per les Collections i un altre per retornar la query utilitzada per les cerques custom.

Per la part de l'Endpoint de les Collections, es passen paràmetres tals com el país, el tipus de Collection (ja que s'utilitzen també en altres tipus d'ordenacions), etc. La resposta a aquesta crida és el fitxer comprimit JSON que prèviament s'havia calculat amb la pipeline de la Collection.

Per l'altra banda, l'Endpoint de la query del Search Engine, s'utilitzarà bàsicament per retornar la petició GET que haurà d'enviar la web (quan un usuari faci una cerca custom) a la Search API d'Elasticsearch.
Els paràmetres que es passen és el text que l'usuari ha introduït i el país en el qual es troba, així, es retorna la petició apuntant a l'índex corresponent del país.

9.2.6.2. Autocompletat

En el cas de l'autocompletat, per tal que l'equip de web pugui mostrar suggerències mentre un usuari està escrivint al cercador, hi hauran de guardar a la memòria del

servidor el fitxer que es retorna en cridar a l'Endpoint de les Collections, el qual és el mateix que s'utilitza per a la Collection del Search Engine. Tot i que, com el paràmetre del tipus de Collection és diferent, el path per a buscar el fitxer a S3 és diferent i serà el que s'ha creat per l'autocompletat. Per tant, el fitxer que retorna com a resposta serà la llista de suggerències ordenada de forma decreixent pel Score.

9.2.7. Jobs automàtics de Kubernetes

Com els ítems publicats poden anar variant, igual que les cerques custom dels usuaris i els valors de les features (sobretot les relacionades amb les estadístiques dels usuaris), cal anar actualitzant de forma constant els documents indexats a Elasticsearch, posteriorment les Collections del Search Engine i les Collections de l'autocompletat.
Per tant, un cop al dia se sobreescriuen completament els fitxers de Features, Scores i Collections, així com els índexs per cada país del Search Engine.
A més, 6 cops al dia, es realitza també una execució incremental per si s'han afegit nous ítems o les cerques pels usuaris han canviat notòriament.

Generalment, el que executen els jobs són workflows. Un exemple de workflow pel Search Engine seria una pipeline, que, per fases, executa les següents pipelines:

- Fase 1: Features.
- Fase 2: Score.
- Fase 3: Pujada dels documents a Elasticsearch.
- Fase 4: Collections.

9.3. Selecció de les dades a utilitzar

En aquesta secció, es parlarà sobre les taules de la BD de Tiendeo que s'utilitzaran per definir els camps que contindrà un document. Bàsicament, es tindrà en compte els textos que un usuari podria introduir per a fer una cerca en la web i s'indexaran a Elasticsearch aquests camps de text en forma de document, ja que, en el cas del cercador, un document en Elasticsearch representa un ítem.

9.3.1. Taules a utilitzar

Per tal de veure com es defineix un ítem, s'observen les següents taules:

- *Articles.* És una taula la qual conté tota la informació relacionada amb els articles.

- *Sales.* És una taula la qual conté la informació relacionada amb les sales.

- *Products.* Per a donar més definició a la donada prèviament, un producte és una manera de classificar un ítem en una jerarquia. És a dir, cada producte es troba dins d'un arbre jeràrquic (el qual Tiendeo anomena *Breadcrumb*). Per exemple, un article que sigui un ordinador qualsevol, tindrà associat un producte en l'arbre jeràrquic, que en aquest cas serà "*PC sobremesa",* i de forma jeràrquica ascendent, tindrà "*ordenadores"*, "*informática"* i, per últim, "*informática y electrónica"* (figura 18).

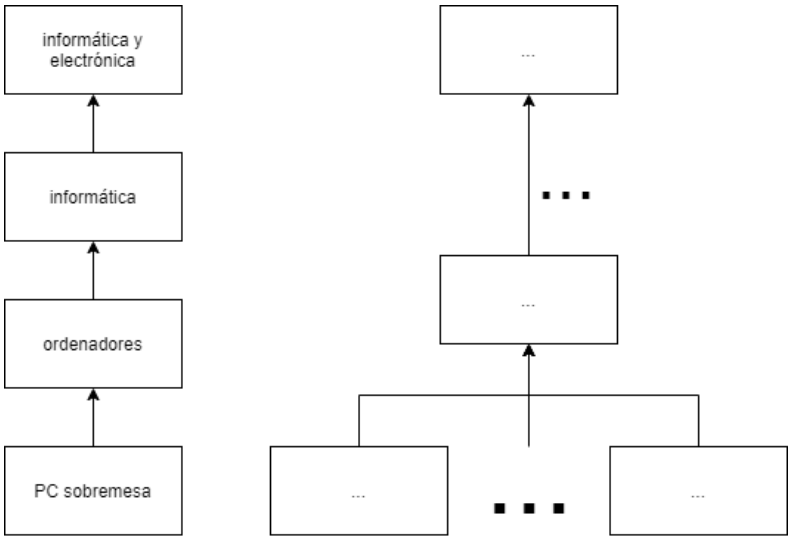


Figura 18: Mostra l'estil jeràrquic dels productes.
Font: Elaboració pròpia.

También podria donar-se el cas que un producte tingués dos pares. Sigui, per exemple, el producte "*vestidos niña",* té de pares "*ropa niña"* i "*vestidos".*

- *Brands.* Aquesta taula té tota la informació relacionada amb les marques. Per exemple, l'article "iPhone 12", té com a marca "Apple".

- *Accounts.* Aquesta és una taula la qual conté informació relacionada amb minoristes (retailers). És a dir, si es pren el mateix exemple de l'iPhone 12, un possible minorista podria ser: "Carrefour".

- *Synonyms.* És una taula d'on es poden extreure sinònims posats a mà de cada Producte, Retailer, Marca., etc. Un possible exemple podria ser, del producte "cerveza", els sinònims: "cerveza de frambuesa" i "cerveza de manzana".

Clarament, totes les taules prèviament esmentades es troben particionades per país, doncs no es tenen els mateixos articles a Espanya que a França.

9.3.2. Camps que defineixen un document

Per tal de veure quins camps defineixen un document, es veuran les taules que s'utilitzaran per generar-los.
Primerament, es té la taula anomenada *articles* i disposa de les següents columnes d'interès (es mostraran només les columnes que siguin útils, ja que hi ha d'altres que s'han inclòs, sigui per debugging[22] o per camps d'informació addicional):

- Id: és un identificador numèric únic per article. Aquest s'afegirà com a field al document, tot i que no serà utilitzat a l'hora de fer la cerca (ja que, clarament, un usuari no farà una cerca d'un id d'article, sinó, més bé, de com s'anomena aquest article), però serà útil com a desenvolupadors per a tenir informació de quin article se n'està fent referència.
- Text: és el nom de l'article. Un usuari pot fer una cerca explícitament del nom d'un article, que podria ser, per exemple, "bicicleta para niños".
- AdditionalText: és una breu descripció de l'article generat per OCR.
- ProductId: és un identificador numèric únic per producte. Igual que l'id d'article. S'utilitzarà per debugging i fer un join amb la taula de *Products.*
- BrandId: és un identificador numèric únic per marca. Igual que abans, s'utilitzarà per debugging i per fer join amb la taula *Brands.*
- RetailerId: és un identificador numèric únic per minorista. S'usarà per debugging i fer join amb la taula *Accounts.*

De la taula *Products,* tot article pot tenir un producte relacionat (ProductId), per tant, s'haurà de fer un merge de les taules respecte a aquest id de producte, i, com aquest es troba dins d'un arbre jeràrquic, pot tenir un o dos pares (a partir d'ara, si un producte tingués un segon producte pare, s'anomenarà adoptiu) i, aquests pares, poden tenir pares i així successivament.
Per tant, tota aquesta informació és necessària guardar-la per a un article, perquè un usuari pot estar buscant "*informática"* i, per tant, a aquest li sortiran articles d'ordinadors, tauletes, etc., ja que són els fills del producte "informática". Tot i això, s'ha de mantenir un nivell màxim d'alçada, i aquesta serà de 6, ja que és una constant definida dins el codi de Tiendeo i no hi ha molts productes els quals tinguin 6 pares com a màxim per sobre seu.
Per tant, per cada nivell de producte, s'agafa la següent columna:

- Name: és el nom d'un producte.

Obtenint així el nom del producte base, 5 nivells de noms de productes i 5 de productes adoptius.

Per la part de les taules *Brands* i *Accounts*, es tracten igual, doncs, per tot article, es pot tenir una marca (BrandId) i/o un minorista (RetailerId) relacionat a aquest. Tot i que, en el cas del Retailer, s'afegeix la Categoria d'aquest (un exemple de categoria

[22] Procés per a identificar i corregir possibles errors de programació[36].

seria, per "*Lidl",* "Supermercados", per "*Media Markt",* "Electrónica"...). Més endavant s'explicarà la seva funcionalitat.
Fent el merge de les dues taules, respecte a aquests Id's, s'agafarà, de les dues taules, la següent columna:

- Name: és el nom de la marca o minorista. Un usuari pot cercar per aquests noms.

- RetailerCategoryIds: és l'id (o ids) de la categoria a la qual el minorista pertany.

També, de la taula *Synonyms*, s'extreuen tots els sinònims que estan relacionats amb els Retailers, Categories, Productes i Marques. Aquesta informació, en el següent apartat, s'explicarà el seu ús específic.

Per últim, com a camp d'importància, també s'afegirà l'Score calculat per cada article (com bé ja s'ha explicat anteriorment el seu càlcul).

Com també es volen afegir les sales, el procés és exactament igual al realitzat amb els articles, és a dir, els camps que té la taula *Articles,* també els té la taula *Sales*, i, per tant, el procediment és el mateix. Llavors, ja s'estarien afegint les dues entitats que es poden cercar amb el cercador.

Per tal de veure un exemple de com seria un document amb els fields que s'han definit, tenim la figura 19. En aquesta, cada fila correspon a un field diferent. Es pot observar que es tracta d'un l'article (camp ItemType) anomenat "*Cerveza ALHAMBRA Reserva 1925"* (ArticleText). A més, el seu producte és "*cerveza sin alcohol"* (ProductLevel0), té 4 pares per sobre seu (ProductLevel1 - 4) i succeeix que cap d'aquests en té dos pares (cap adoptiu o AdoptiveProduct com es veu a la figura). La marca corresponent a aquest article és "*Alhambra"* (BrandName) i el minorista és "*Carrefour"* (RetailerName).

```
"ItemId": "ARTICLE-23302981",
"Score": 0.522614472,
"ProductId": 1508,
"BrandId": 556,
"ArticleText": "Cerveza ALHAMBRA Reserva 1925 ",
"RetailerId": 35,
"ArticleId": 23302981,
"SaleId": 0,
"ItemType": "ARTICLE",
"TagBox": true,
"SimilarityId": "7231C17E-CD7F-F178-1F68-D5D301D596EC",
"AdditionalText": "",
"Active": true,
"ModelType": "ArticleSearchEngine",
"ModelName": "ContentBased_v1",
"Country": "ES",
"ProductLevel0": "cerveza sin alcohol",
"ProductLevel1": "cerveza",
"AdoptiveProductLevel1": "",
"ProductLevel2": "bebidas alcohólicas",
"AdoptiveProductLevel2": "",
"ProductLevel3": "bebidas",
"AdoptiveProductLevel3": "",
"ProductLevel4": "supermercados",
"AdoptiveProductLevel4": "",
"ProductLevel5": "",
"AdoptiveProductLevel5": "",
"ProductLevel6": "",
"AdoptiveProductLevel6": "",
"BrandName": "Alhambra",
"RetailerName": "Carrefour",
```


Figura 19: Es pot observar un article i tots els camps afegits.
Font: Elaboració pròpia.

9.4. Definició dels analyzers

Un analyzer en Elasticsearch, és la forma en la qual es processarà un text, amb la finalitat que, si un usuari cerca alguna de les següents paraules: “Librería”, “librería”, “libreria”, “librerías”, “librerias”, etc., acabi trobant sempre el mateix resultat.

Per a obtenir aquest objectiu, Elasticsearch té diferents filtres implementats per aplicar als textos i també s’aplicaran tècniques de processament del llenguatge natural (NLP). Aquests filtres que es mostren a continuació s’apliquen de forma seqüencial als camps de tipus text dels documents indexats a Elasticsearch:

- Lowercase: Per a passar a minúscula tota mena de text. D’aquesta manera, no es distingeixen paraules diferents depenent si es troba en majúscules o minúscules.
- Asciifolding: Per treure tot signe de puntuació (“librería => “libreria”). Així, si un usuari comet alguna falta d’ortografia, no cal preocupar-se.
- Keywords: És una llista de paraules que, si una paraula d’aquestes fa un match en un text, aquesta es mantindrà sense modificar pels filtres següents. Uns exemples pot ser “Mercadona”, “Lidl”, ...

- Stopwords: És una llista de paraules de preposicions, articles determinants, etc. que, si una paraula d'aquestes fa un match en un text, s'elimina del text. D'aquesta manera s'aconsegueix eliminar text no important i que pot generar mals resultats a l'hora de fer una cerca, perquè si es busqués "arroz con tomate", potser podria sortir "pasta con atún", pel fet de tenir una paraula (la qual és una preposició) coincident.
- Stemmer: És un mètode per a reduir cada paraula a la seva arrel. Per exemple, faria una regla com aquesta: ("libreria", "librerias") => "libreri". D'aquesta forma, s'aconsegueix que, ja sigui una paraula en singular, plural o variant d'una altra, acabi parant a aquesta mateixa. A més, el nombre de paraules que es troben en l'índex invertit[23] d'Elasticsearch és reduït considerablement per tal d'obtenir un temps de cerca menor.

Clarament, l'usuari final no veurà la paraula "libreri", sinó que ell veurà la forma original d'aquesta, "Librería".

A més, tot analyzer té una forma de tokenitzar les paraules, és a dir, la forma de dividir una frase en paraules. Per exemple, la frase: "Arroz con tomate", s'hauria de segmentar en "Arroz", "con", "tomate" (si no s'apliqués cap filtre previ) i la forma en la qual es farà és mitjançant un standard de segmentació de text unicode[24] (ja implementat en Elasticsearch).

9.4.1. Configuració per a cada idioma

Com cada idioma és diferent entre si, s'han de fer configuracions extres per alguns en específic, com eliminar els apòstrofs en el Francès i Italià, afegir alguns plugins d'Elasticsearch pels idiomes tals com el Japonès, Coreà, etc., ja que la majoria dels filtres anteriors no serveixen per llenguatges que no provenen del llatí, així com canviar la forma de tokenitzar, etc. Per tant, si fos un llenguatge especial, s'haurien d'aplicar diferents filtres i/o tokenizers.

Llavors, es definiran dos analyzers, un amb els filtres esmentats prèviament i un altre aplicant un filtre extra, que serà afegint els sinònims.

[23] Estructura de dades que guarda, per tota paraula trobada entre tots els documents, en quin document és trobat. Obtenint, llavors, una llista de: paraula-llista de documents on es pot trobar.
[24] Documentació a: https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis-standard-tokenizer.html

```
"search_engine_analyzer_synonym": { -
  "filter": [ -
    "lowercase",
    "asciifolding",
    "ES_synonym",
    "ES_keyword",
    "ES_stop",
    "ES_stemmer"
  ],
  "tokenizer": "standard"
},
"search_engine_analyzer": { -
  "filter": [ -
    "lowercase",
    "asciifolding",
    "ES_keyword",
    "ES_stop",
    "ES_stemmer"
  ],
  "tokenizer": "standard"
}
```

Figura 20: Mostra els dos analyzers per Espanya.
Font: Elaboració pròpia.

```
"RetailerName": { -
  "type": "text",
  "copy_to": [ -
    "ArticleProductBrandRetailer"
  ],
  "analyzer": "search_engine_analyzer"
},
"RetailerNameSynonym": { -
  "type": "text",
  "copy_to": [ -
    "ArticleProductBrandRetailerSynonym"
  ],
  "analyzer": "search_engine_analyzer_synonym"
},
```

Figura 21: Mostra dos camps en els quals s'apliquen analyzers diferents.
Font: Elaboració pròpia.

Com s'observa en la figura 20, l'única diferència entre els dos analyzers, és el filtre de "ES_synonym", el qual farà una expansió de paraules si troba un text dins d'una llista de sinònims. Vegeu el següent exemple:
Suposant un exemple, on, en el camp RetailerName (com es veu a la figura 21) amb l'analyzer sense sinònims, s'insereix el text: "El Corte Inglés", s'obtindrien els següents tokens: "cort", "ingl". Els passos realitzats han estat:

1. Lowercase: "el corte inglés".
2. Asciifolding: "el corte ingles".

3. Keywords: “el corte ingles”.
4. Stopwords: “corte ingles”.
5. Stemmer: “cort ingl”.
6. Tokenització: “cort”, “ingl”.

Mentre que, si aquest mateix text, s’introdueix amb l’analyzer amb sinònims (en el camp RetailerNameSynonym) i se suposa que “El Corte Inglés” té de sinònim: “Telecor”, s’obté el text següent:

1. Lowercase: “el corte inglés”.
2. Asciifolding: “el corte ingles”.
3. Synonyms: “el corte ingles telecor”.
4. Keywords: “el corte ingles telecor”.
5. Stopwords: “corte ingles telecor”.
6. Stemmer: “cort ingl telecor”.
7. Tokenització: “cort”, “ingl”, “telecor”.

La paraula “telecor” ha sortit directament en minúscules i aplicant asciifolding (tot i que en aquest cas no tenia cap signe de puntuació), ja que els sinònims s’han d’indexar amb els filtres previs aplicats per cada país (com es troba en la posició 3 dels filtres, s’ha d’indexar fent lowercase i asciifolding) a Elasticsearch, utilitzant NLP.

Per tant, l’aplicació pràctica de l’analyzer sense sinònims és ajuntar tot text semblant des de l’arrel de la paraula per a poder fer match entre la cerca d’un usuari i un document de forma més senzilla i la de l’analyzer amb sinònims es veurà en el següent subapartat.

9.5. Creació de la query de cerca

La query de cerca és el cos de la crida a la Search API[25] d’Elasticsearch.
Aquesta, és utilitzada per retreure documents que compleixen certes condicions. Per tant, si es transporta al Search Engine que es vol desenvolupar, l’objectiu és que, a partir d’un text, retorni ítems ordenats segons prioritat.
Llavors, en els següents subapartats, s’explicarà les diferents queries utilitzades, certs valors pels camps en els quals es faran les cerques i es mostrarà la query final en format JSON.

[25] Documentació Search API. Disponible a: https://www.elastic.co/guide/en/elasticsearch/reference/current/search-search.html

9.5.1. Boolean Query

Per tal de cercar els ítems necessaris, s'utilitzarà la Boolean Query[26]. Aquesta, aporta un cert grau d'utilitat el qual és necessari per un Search Engine:

- Es pot posar una condició obligatòria per a retornar un document determinat. Per exemple, a partir d'un text, com a mínim, alguna de les paraules (tokens) apareixin en el document que es vol retornar. És a dir, si es cerca "*Televisor 50 pulgadas*", es vol que si es retorna un ítem, aquest, per exemple, es digui "Televisor Samsung".
  Aquesta funció s'aconsegueix amb la Must Query de la Boolean Query.

- A més, es poden posar condicionals, que, en el cas que es compleixi certa condició, farà que un document donat, el seu score amb el qual estarà rankejat (posicionat) per Elasticsearch sigui boostejat (incrementat).
  Un exemple d'ús podria ser el cas anterior. Suposant que es fa la cerca de "*Televisor 50 pulgadas*", es pot posar la condició que els documents amb el text exacte de la cerca estiguin millor rankejats que no pas uns altres que tenen alguns tokens de la cerca. Per exemple, un ítem anomenat "Televisor 50 pulgadas" estarà millor posicionat que un anomenat "Televisor 24 pulgadas".
  També pot ser utilitzat per boostejar l'score generat per Elasticsearch directament a partir de cert valors (com pot ser l'Score que s'ha calculat prèviament en les pipelines).
  Això s'aconsegueix amb la Should Query de la Boolean Query.

9.5.1.1. Must Query

Per tant, l'única condició obligatòria que es posarà a la query serà que com a mínim un token de la cerca de l'usuari analitzat amb l'analyzer sense sinònims (ja que no volem expandir tokens, ja que l'usuari vol cercar un producte en específic) faci match en una llista de camps de text, això s'aconsegueix amb la Multi-match query. Aquests camps han estat indexats a Elasticsearch (com s'ha explicat prèviament) i duplicats, ja que, uns tindran aplicats l'analyzer sense sinònims i l'altre amb sinònims.
D'aquesta manera, com a la Multi-match query es pot posar un cert grau d'importància als camps, els camps analitzats amb l'analyzer amb sinònims se li donarà menys, ja que així s'aconsegueix que, per una cerca, els productes relacionats a aquesta cerca (els quals han estat expandits pels sinònims) surtin per la part final de la llista de resultats i els més rellevants a l'inici (els quals no han estat analitzats per l'analyzer amb sinònims). Per exemple, si se cerca "Patatas fritas",

[26] Documentació de la Boolean Query. Disponible a: https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-bool-query.html

que primer surti un ítem amb títol “Patatas fritas” i, després, un amb títol “Pringles”, ja que és un sinònim de “Patatas fritas”.

Per tant, en els camps en els quals es volen fer les cerques són els següents (s’hauran de duplicar per tenir amb sinònims i sense):

- Text
- AdditionalText
- Name (de la marca)
- Name (del retailer)
- Productes del nivell 0 fins al 6
- Productes adoptius del nivell 1 fins al 6

També, se sap que, a menys camps cerquis en la Multi-match query, més ràpida serà la cerca[28]. Per tant, com per cada camp se li ha de donar una importància, per tal de reduir el temps de cerca, s’ajuntaran en un únic camp de text els camps amb la mateixa importància (com al cap i a la fi s’hi estan cercant tokens, en aquest camp nou creat seria la concatenació del text d’aquells camps de text amb la mateixa importància).
A continuació es té, per tant, els camps en els quals es farà la cerca, ordenats de major a menor importància, agrupats per reduir el temps de cerca i duplicats per tenir-los analitzats amb diferents analyzers:

1. *ArticleProductBrandRetailer*: És el camp creat de la concatenació de text del títol de l’article, el nom del producte de nivell 0, el nom de la marca i el nom del retailer.
2. *Level1Product*: Camp creat de la concatenació del nom del producte de nivell 1 i el producte adoptiu de nivell 1.
3. *AdditionalText*.
4. *Level2Product*: Idem. a Level1Product, però de nivell 2.
5. *Level3Product*: Idem. a Level1Product, però de nivell 3.
6. *Level4Product*: Idem. a Level1Product, però de nivell 4.
7. *Level5And6Product*: Camp creat de la concatenació del nom del producte de nivell 5 i 6, així com el producte adoptiu 5 i 6.
8. *ArticleProductBrandRetailerSynonym.*
9. *Level1ProductSynonym*.
10. *Level2ProductSynonym*.
11. *Level3ProductSynonym*.
12. *Level4ProductSynonym*.
13. *Level5And6ProductSynonym*.
14. *Level6ProductSynonym*.

Com es pot observar, el camp AdditionalText és l’únic que no té el camp duplicat amb l’analyzer amb sinònims, ja que, en ser una gran quantitat de text detectat amb OCR, pot fer una gran expansió errònia de tokens i, a més, afectar a la performance.

També, el camp amb més importància és *ArticleProductBrandRetailer*, perquè un usuari sempre farà alguna cerca fent referència a alguns d'aquests camps. A més, els nivells de producte, a mesura que s'hi va pujant, va perdent importància, ja que aquest es va fent més general. Com es pot observar, el nivell 5 i 6 es troben junts, ja que en tenir la més baixa importància, aquest s'han ajuntat.
A més, cal destacar que el camp AdditionalText es trobi entre els valors *Level1Product* i *Level2Product*. Això és degut al fet que s'ha estimat que la importància del producte pare (i adoptiu) de nivell 1 és més important que una descripció en la qual pot haver-hi paraules errònies. I, a més, el producte de nivell 1 és bastant semblant al producte de nivell 0, i, per tant, té una bona descripció de l'ítem. Mentre que el nivell 2 perd certa definició, la qual AdditionalText, com a descripció que s'ha detectat, la té.
Per últim, cal aclarir que, en el cas que es tracti del cas de la Collection de tipus producte, la importància dels nivells de producte i els seus sinònims creixen. En canvi, si es tracta de la Collection de marques, la importància del nom de la marca (així com el seu sinònim), creix.

Com es pot observar a la figura 22, s'observa la millora del temps d'execució de cinc cerques diferents. On la barra grisa seria el temps d'execució sense agrupar camps, i, la barra blava, agrupant-los.

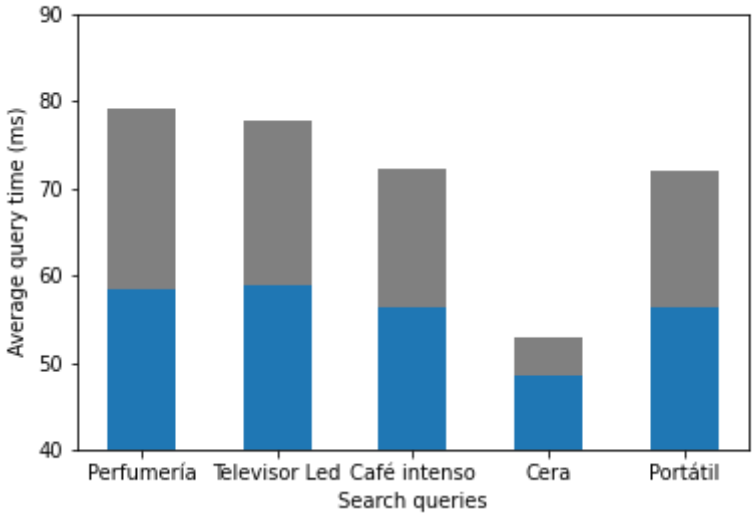


Figura 22: Mostra el temps d'execució per cinc cerques diferents.
Font: Elaboració pròpia.

Per últim, com el que es volia aconseguir era que un mínim d'un token de la cerca de l'usuari faci match en els camps definits prèviament, el paràmetre *operator* de la Multi-match query serà "OR".

9.5.1.2. Should Query

Com a condicions que faran posicionar millor a certs ítems, tenim:

- Una multi match query, exactament igual a la definida en la Must Query, però el paràmetre *operator* està posat a "AND", ja que farà, que si tots els tokens generats per la cerca de l'usuari es troben en alguns dels camps d'un ítem, l'score que genera Elasticsearch per ordenar els ítems serà incrementat per aquell ítem en específic.

- Per donar més importància a alguns ítems en específic (l'Score calculat amb les pipelines), s'utilitzarà la Rank feature query[27].

- També, només pel cas de les Collections per les paraules precalculades dels productes, s'afegirà una Multi-match query, igual que l'esmentada a la Must Query, per cada fill, nét, etc. (no s'aplica per les marques, ja que no hi ha una jerarquia de marques definida) que un producte tingui. Això és degut al fet que, si un usuari cerca "*Móviles*", el que s'espera són mòbils de Samsung, iPhones, etc., però, com per arribar a aquest producte fulla, ha de passar per 3 nivells de productes (Móviles -> Smartphones -> Smartphones Samsung -> …), el text "Móviles", només es trobarà, per un mòbil Samsung qualsevol, en el producte de nivell 4 en endavant.
  Llavors, com s'ha explicat anteriorment, un producte de nivell 4 tindrà poca importància, i, per tant, tot i trobar-se una sola coincidència, no serà retornat com a resultat (ni tan sols cap al final d'una possible llista), ja que l'score que li donarà Elasticsearch no serà suficient per passar un threshold, el qual s'explica amb més profunditat al següent subapartat.

- Per últim, per les dues Collections de les paraules precalculades, s'afegeix una condició que és que si la categoria del producte (o de la marca, depenent del tipus de Collection) és igual a la categoria del retailer d'un ítem en específic, el posicionament d'aquell ítem es veurà millorat. S'utilitzaria una Term query[28] i aquesta condició pretén millorar que si es fa una cerca precalculada, com per exemple: *"Televisores"*, no surtin ítems de l'estil "Televisor de juguete", ja que, amb "Televisores", es vol trobar com a resultat televisors que no siguin joguines. Per tant, com la categoria de "Televisores" seria "Electrónica"

[27] Documentació de la Rank feature query. Disponible a: https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-rank-feature-query.html
[28] Documentació de la Term query. Disponible a: https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-term-query.html

i la del retailer de l'ítem seria "Juguetes y Bebés", aquest ítem no es veurà millor posicionat a diferència d'uns altres.

Com s'ha esmentat, ja que es calculen Collections per tal de reduir al màxim el temps de resposta de la cerca d'un usuari, s'aprofita el grau de coneixement que es té de cada una d'aquestes paraules i es tracta de millorar l'ordenació.

9.5.2. Function Score

Per tal de tallar els articles que es consideren no rellevants a partir de cert valor (threshold), s'utilitzarà la Function score query[29]. Com per cada país, la definició d'un ítem és diferent (és a dir, a Austràlia, per exemple, pot haver-hi més d'un ítem el qual no té assignat cap producte, per tant, la majoria dels camps que defineixen els ítems es troben buits), un valor pot ser correcte per un país i erroni per un altre. És a dir, per Espanya, que la majoria d'ítems es troben ben definits, l'score generat per Elasticsearch serà més gran que el d'Austràlia, ja que es poden trobar més coincidències del text a cercar en tots els camps en els quals es fa la cerca.

Per tant, el mètode per a establir a cada país el seu valor d'score mínim corresponent, ha sigut establir un únic valor inicial a tots els països i, després, anar actualitzant de forma automàtica els valors per a cada país, depenent si la mitjana de la llargada de les Collections per un país determinat és més petita del normal o més gran del normal. Això fa que, si aquesta llargada mitjana és petita, vol dir que l'score mínim establert hauria de ser més petit per tal d'oferir més resultats. Mentre que si es tracta d'una llargada més gran del normal, el valor de l'score hauria de ser més gran per tal d'oferir menys resultats, ja que alguns podrien ser no rellevants.

9.5.3. Query final

Observi's el següent exemple en format JSON de la query resultant en fer una cerca de "Televisores" (suposant que no es tracta d'una paraula precalculada, ja que, sinó, per cada fill que tingués el producte, s'hauria d'afegir una Multi-match query i l'exemple podria fer-se massa llarg):

```
{
  "query": {
    "function_score": {
      "query": {
        "bool": {
          "must": [
            {
              "multi_match": {
```

[29] Documentació de la Function score query. Disponible a: https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-function-score-query.html

```
            "query": "Televisores",
            "fields": [
              "AdditionalText^0.14",
              "ArticleProductBrandRetailer^0.16",
              "ArticleProductBrandRetailerSynonym^0.06",
              "Level1Product^0.15",
              "Level1ProductSynonym^0.04",
              "Level2Product^0.11",
              "Level2ProductSynonym^0.03",
              "Level3Product^0.1",
              "Level3ProductSynonym^0.02",
              "Level4Product^0.09",
              "Level4ProductSynonym^0.02",
              "Level5And6Product^0.08",
              "Level5ProductSynonym^0.01"
            ],
            "fuzziness": "AUTO:6,10",
            "analyzer": "search_engine_analyzer",
            "type": "best_fields",
            "operator": "OR"
          }
        }
      ],
      "should": [
        {
          "multi_match": {
            "query": "Televisores",
            "fields": [
              "AdditionalText^0.14",
              "ArticleProductBrandRetailer^0.16",
              "ArticleProductBrandRetailerSynonym^0.06",
              "Level1Product^0.15",
              "Level1ProductSynonym^0.04",
              "Level2Product^0.11",
              "Level2ProductSynonym^0.03",
              "Level3Product^0.1",
              "Level3ProductSynonym^0.02",
              "Level4Product^0.09",
              "Level4ProductSynonym^0.02",
              "Level5And6Product^0.08",
              "Level5ProductSynonym^0.01"
            ],
            "fuzziness": "AUTO:6,10",
            "analyzer": "search_engine_analyzer",
            "type": "most_fields",
            "operator": "AND"
          }
        },
        {
          "rank_feature": {
            "field": "Score",
            "linear": {},
            "boost": 8
          }
        }
      ]
    }
  },
```

```
    "min_score": 3.5
  }
 }
}
```

Com s'observa i s'ha explicat prèviament, la query està encapsulada per una Function score query ("function_score"), la qual estableix un threshold, que, en aquest cas és 3.5 ("min_score"). Dins d'aquesta, es té la Boolean Query ("bool"), amb la Must Query ("must") i la Should Query ("should").

Dins la Must Query, es té la Multi-match query ("multi_match"), on a dins es pot observar que s'està cercant "Televisores" (en "query") en els camps anteriorment esmentats (la llista de "fields"). També es poden observar els paràmetres de l'analyzer escollit (en "analyzer") i l'operador ("operator"), així com un nou, "fuzziness", que, en aquest cas, fa el següent:

- Si un token generat de la cerca té menys de 6 caràcters, per tal de fer match amb un altre token, els tokens han de ser exactament iguals.
- Si un token generat té entre 6 i menys de 10 caràcters, per a fer match amb un altre token, ha de ser igual o amb una permutació de caràcters (Levenshtein edit distance[30]) de diferència.
- Si un token generat té més o igual de 10 caràcters, per poder fer match amb un altre token, ha de ser igual o amb una o dues permutacions de caràcters de diferència.

Per tant, la seva funcionalitat servirà per si, en el cas que no sigui una paraula precalculada i l'usuari comet una falta d'ortografia, s'intenti, en la mesura del possible, fer match.
Per altra banda, dins la Should Query, es té la Multi-match query amb el seu operador corresponent i la Rank feature query ("rank_feature"), especificant que el valor del camp "Score", s'haurà de multiplicar per 8 (és pel fet de combinar un "boost" de 8 i el paràmetre "linear"), sent això el que se li sumarà al score generat per Elasticsearch. En el cas que fos la query per les Collections de producte, sota de la Rank feature query, es veurien afegides tantes Multi-match queries com fills tingués el producte "Televisores" (en cas que tingués). I, si fos qualsevol de les dues Collections (productes o marques), es veuria una term query com aquest exemple:

```
{
        "term": {
          "RetailerCategoryIds": {
            "value": 4,
            "boost": 1.5
          }
        }
}
```

[30] Nombre d'operacions mínimes requerides per a transformar una cadena de caràcters en una altra[29].

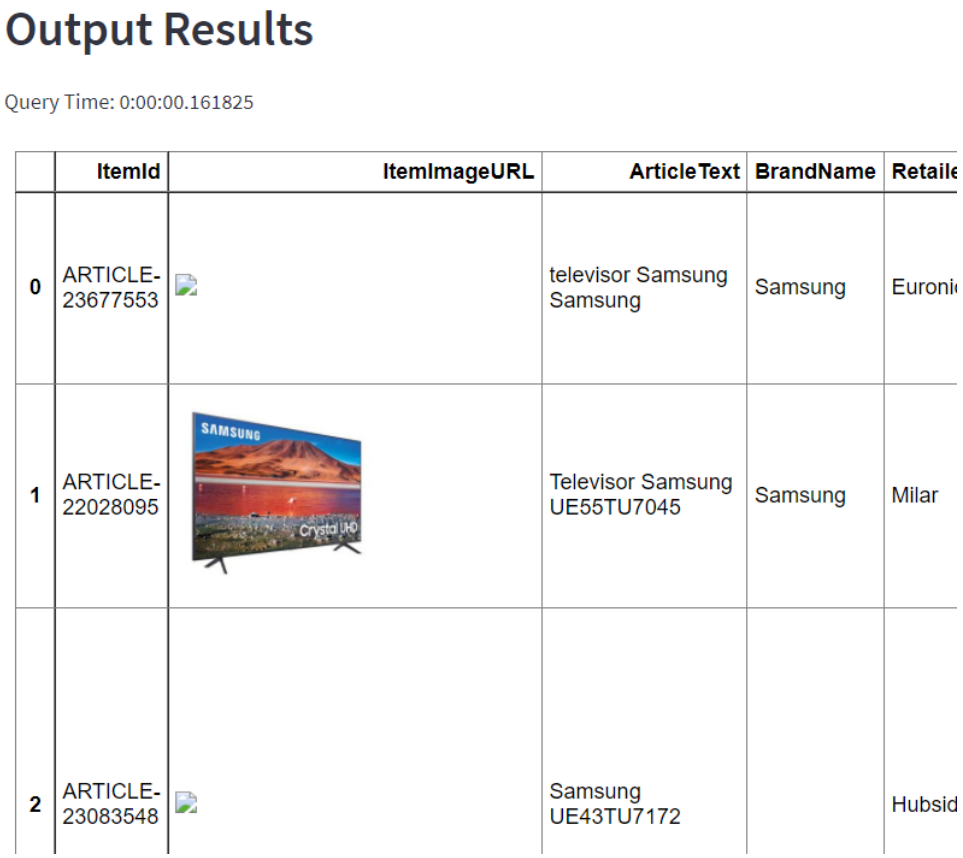
On, suposant que la categoria de “Televisores” fos 4 (que correspondria a *Electrónica*), si la categoria del retailer d’un ítem en específic també fos 4, el seu score d’Elasticsearch es veuria incrementat en 1.5.

Per últim, cal destacar que en la Multi-match query, depenent si es troba dins la Must o la Should Query, el paràmetre “type” té diferents valors. Això és degut al fet que, com dins la Should Query, es vol incrementar l’score dels ítems els quals tots els tokens introduïts en la cerca es trobin en algun dels camps a cercar, si aquests tokens apareixen en més d'un camp, aquest valor sigui incrementat més encara (“most_fields”). Mentre que si es tracta de la Multi-match query, que es troba en la Must Query, com es vol que com a mínim un token de la cerca faci match, igualment es vol donar més importància a aquells ítems els quals tinguin tots els tokens de la cerca, per tant, en comptes de fer una combinació de tots els camps i crear un score (generat per Elasticsearch), s’agafa un score generat pel camp amb millor match dels tokens de la cerca (“best_fields”).

10. Validació dels resultats

A l'hora de poder determinar i poder validar els resultats a escala interna de les iteracions que s'anaven fent del cercador, s'utilitzava l'ajuda d'Streamlit[31] i la mètrica NDCG[32]. I, en el cas de validar els resultats mitjançant el feedback implícit dels usuaris, es realitzava mitjançant gràfiques de CTR.

Streamlit, en el cas de l'equip de DATA, majoritàriament s'utilitza per a poder visualitzar ordenacions de Tiendeo d'una forma més ràpida i dinàmica, sense haver de re-executar processos llargs per poder fer comprovacions. Per tant, com ja es tenien Dashboards[33] amb dades d'altres ordenacions, pel Search Engine s'ha hagut d'adaptar i així també ha pogut ser reutilitzable.
S'observa un exemple en la següent figura 23, en la qual, els únics passos realitzats per a poder comprovar els resultats d'una cerca han estat seleccionar l'ordenació requerida (el Search Engine) i posar el text a cercar.



Output Results

Query Time: 0:00:00.161825

	ItemId	ItemImageURL	ArticleText	BrandName	Retaile
0	ARTICLE-23677553		televisor Samsung Samsung	Samsung	Euroni
1	ARTICLE-22028095		Televisor Samsung UE55TU7045	Samsung	Milar
2	ARTICLE-23083548		Samsung UE43TU7172		Hubsid

Figura 23: Mostra un Dashboard d'Streamlit per la cerca "Televisores Samsung". Font: Elaboració pròpia.

[31] Llibreria de Python que facilita la creació d'aplicacions web per l'aprenentatge automàtic i la ciència de dades.
[32] Normalized Discounted Cumulative Gain. Mètrica per mesurar l'efectivitat de Search Engines i semblants[30].
[33] Interfície gràfica d'usuari que ajuda a mostrar dades i informació de forma interactiva i fàcil.

10.1. Qualitat dels resultats

Per tant, per tal de comprovar la qualitat i validesa dels resultats, primerament es veurà aplicada la mètrica NDCG a les 20 cerques precalculades més utilitzades (ja que així es pot comparar amb el cercador anterior, perquè les cerques custom no és una característica que es pugui comparar), utilitzant els vint primers resultats obtinguts amb el cercador antic i el Search Engine desenvolupat (en el cas que hi hagi menys, s'utilitzaran els que hi hagi), i, seguidament, els gràfics de CTR resultants dels usuaris en interactuar amb el cercador.

10.1.1. Normalized Discounted Cumulative Gain

NDCG, d'inici a fi de la llista a validar mesura la qualitat de cada cerca amb una ponderació diferent i tenint en compte que, cada llista de totes les comparades, poden tenir mides diferents. Aquesta es defineix com:

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

Figura 24: Mostra el càlcul DCG d'una llista.
Font: https://en.wikipedia.org/wiki/Discounted_cumulative_gain

$$\mathrm{IDCG_p} = \sum_{i=1}^{|REL_p|} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

Figura 25: Mostra el càlcul ideal de DCG d'una llista.
Font: https://en.wikipedia.org/wiki/Discounted_cumulative_gain

$$\mathrm{nDCG_p} = \frac{DCG_p}{IDCG_p}$$

Figura 26: Mostra el càlcul nDCG d'una llista.
Font: https://en.wikipedia.org/wiki/Discounted_cumulative_gain

Per tant, consisteix a, en cada posició de la llista d'ítems retornada, valorar, de forma manual, entre el rang d'enters [0,2], sigui 0 res rellevant a la cerca, 1 una mica rellevant i 2 molt rellevant. D'aquesta manera es formarà la llista rel.
Seguidament, es calcula el Discounted Cumulative Gain, computant el sumatori de la divisió que es mostra a la figura 24, on es té en compte cada posició de la llista rel calculada de forma manual.
A continuació es fa el mateix càlcul que amb DCG, però ara la llista rel està ordenada de forma decreixent (figura 25), fet que provoca que, de la llista retornada, sigui la millor solució. Per últim, com s'observa a la figura 26, es fa la divisió entre DCG i IDCG (llista ordenada). Vegeu el següent exemple:

rel = [2,2,1,2,0,1]
REL = [2,2,2,1,1,0]
Sigui rel la llista de rellevància, i REL la llista rel ordenada de forma que sigui la millor solució possible (que és, ordenar de forma decreixent la llista rel).
Es té que DCG = 3,0 + 1,893 + 0,5 + 1,292 + 0,0 + 0,356 = 7,041
IDCG = 3,0 + 1,893 + 1,5 + 0.431 + 0,389 + 0,0 = 7,210
nDCG = 7,041/7,210 = 0,977.
Com es pot veure, en ser un valor normalitzat, com a màxim nDCG pot tenir un valor d'1 (i un mínim de 0), i, a més pròxim d'1, millor es considera la llista de resultats. En aquest cas, està pròxim a aquest valor, ja que la majoria de bons resultats estan a l'inici de la llista.

Per tant, un cop realitzada la valoració manual entre 0 i 2 (consultada i validada pel Product Owner) de cada un dels valors de les llistes de les 20 paraules precalculades més cercades pel cercador antic i el nou, així com calculat el respectiu NDCG, s'obté la següent figura 27:

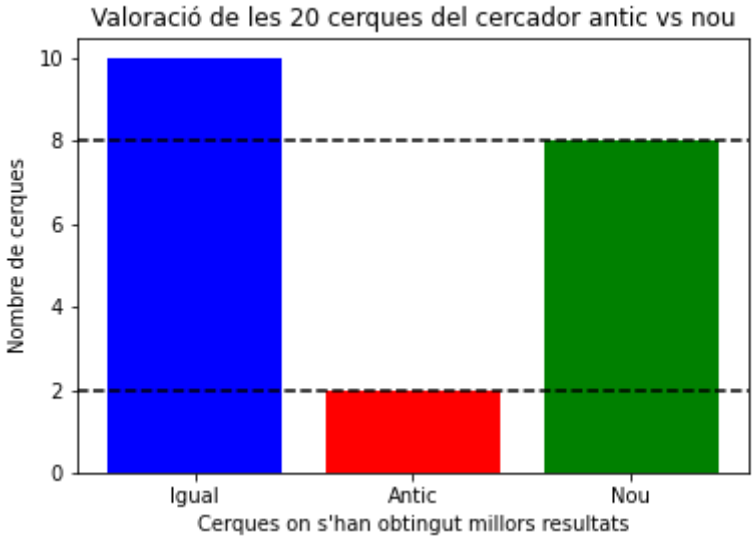


Figura 27: Mostra el gràfic de barres del NDCG per les 20 cerques.
Font: Elaboració pròpia.

Com s'observa, la meitat de les vint cerques s'han quedat igual. Tot i això, una gran part de les cerques (8), respecte a l'antic cercador (2), han estat millorades.
Entrant més en detall, hi havia cerques tal com "ordenador portátil" la qual ha estat una mica empitjorada i això és degut a la paraula combinada "portátil", ja que apareixen algunes fundes per ordinadors portàtils i no és tan rellevant, però com la seva categoria és la mateixa que per ordinador portàtil, no ha baixat en el seu posicionament (això és degut a l'explicat en la Should Query). Al contrari ha passat amb la cerca "cámara digital", ja que, si no s'apliqués la condició de categoria, sortia molta varietat amb el text "digital".

Llavors, aquesta mètrica serveix per a poder donar el vistiplau que el cercador funciona, perquè, com s'ha observat a la figura 27, ens indica que només s'estan empitjorant dues de les vint cerques, amb quasi la meitat millorant-les. Tot i això, el potencial del cercador per les paraules precalculades és que, a diferència del cercador antic, pot trobar ítems que estan malament taguejats, sigui per errors humans o generats per bots, i això no es veu reflectit al NDCG, però sí al feedback dels usuaris, perquè tindran més ítems com a resultats, mentre que amb l'antic, aquesta llista de resultats es mantenia gairebé constant.
A més, tampoc es contempla el fet que un usuari pot cercar una paraula que no estigui precalculada. Prèviament, si un usuari volia cercar un text que no estigués precalculat, o bé se li retornava una pàgina sense resultats o bé havia de cercar per la paraula precalculada més pròxima a la seva cerca, i això podia generar que la seva cerca no fos exitosa, baixant el CTR. Mentre que ara, a l'oferir més possibilitats a l'usuari, és més probable fer augmentar el CTR i les possibles navegacions dels usuaris amb el cercador, tot i que això es veurà al següent subapartat.

10.1.2. Click Through Rate

En aquest apartat es veurà l'impacte en els usuaris en afegir el nou cercador mitjançant els seus clics, fent la mitja entre tots els països. Vegeu la següent figura:

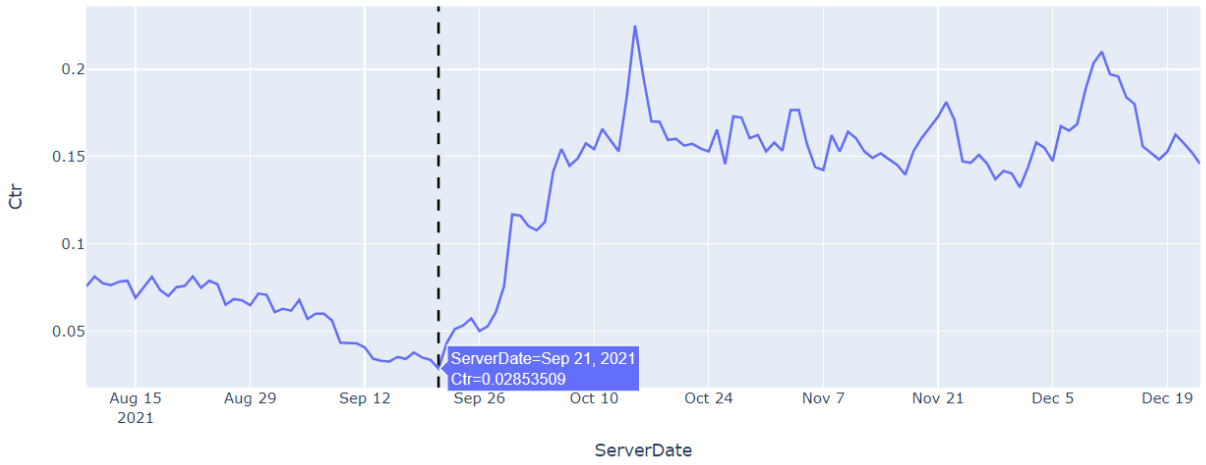


Figura 28: Mostra la distribució del CTR durant el temps.
Font: Elaboració pròpia.

Si s'observa la figura prèviament al dia 21 de setembre (previ a la línia discontinua negre vertical), és quan el cercador antic estava activat. Com s'observa, la tendència del CTR era a anar decreixent fins a arribar al mínim d'un CTR de 2.85%. Això és degut al fet que la majoria d'ítems que s'anaven afegint a Tiendeo era per bots, i, per tant, hi havia de mica en mica un increment de taguejos d'ítems incorrectes, fet que feia penalitzar molt al cercador antic.

A partir del dia 21, que és quan va començar a estar operatiu el nou cercador, veiem que la tendència ha sigut creixent fins a mitjans d'octubre (un mes després), a causa de, com s'especifica a la planificació, s'anava iterant i millorant el cercador, fins a arribar a un CTR estable per sobre del 15%.
Es poden veure tres pics de CTR, un al voltant del dia 15 d'octubre, un altre al 22 de novembre i un últim al 10 de desembre. Tots aquests, són deguts, respectivament, a una campanya de Carrefour molt potent, a la setmana del Black Friday i la precampanya de Nadal.

A continuació, vegeu la figura 29:

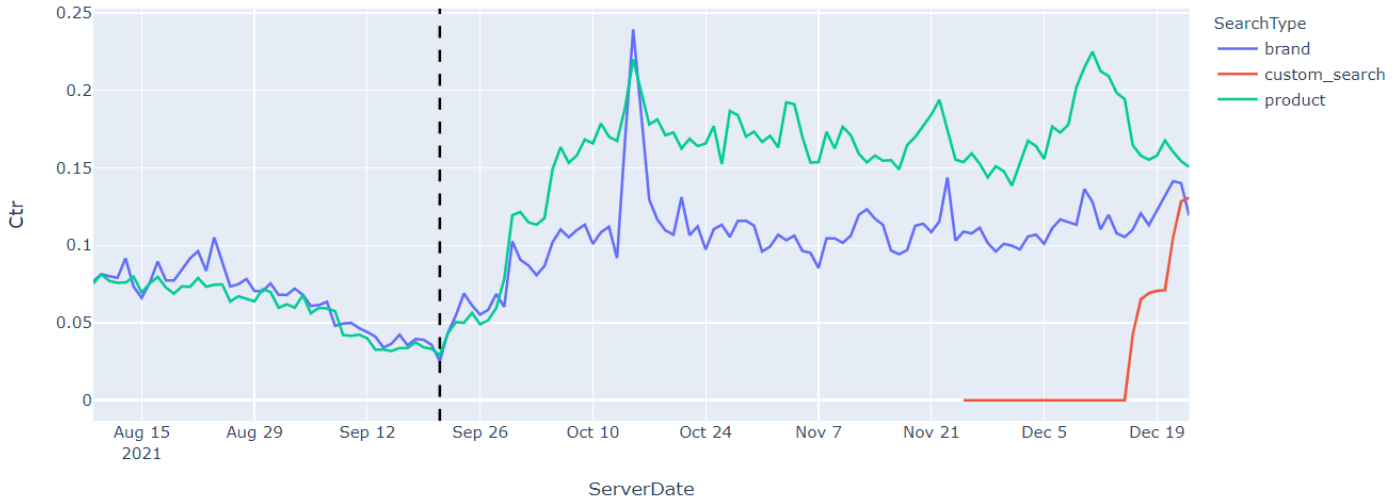


Figura 29: Mostra la distribució del CTR durant el temps, separant per tipus de cerca.
Font: Elaboració pròpia.

Com es pot observar a la figura 29, el tipus de cerca (SearchType) que correspon a brand i product, són les Collections, mentre que custom_search correspon a les cerques custom. A més, com s'ha explicat prèviament, al voltant del dia 15 d'octubre, hi ha un augment significatiu del CTR, degut a les marques relacionades amb el retailer Carrefour.
També es pot apreciar que el CTR de la Collection dels productes és millor al de les marques, ja que tenim més camps que defineixen els productes (tots els nivells dels productes) i només un per les marques.
A més, en el cas de les cerques custom, es van activar el dia 23 de novembre, però va haver-hi un problema en la recaptació de les dades de les obertures dels usuaris i, fins al dia 15 de desembre, de mica en mica es van anar activant l'enviament d'aquestes dades fins a arribar a un possible CTR estable del 15%.
Tot i això, en la següent figura es pot veure la distribució de les navegacions dels usuaris amb el cercador:

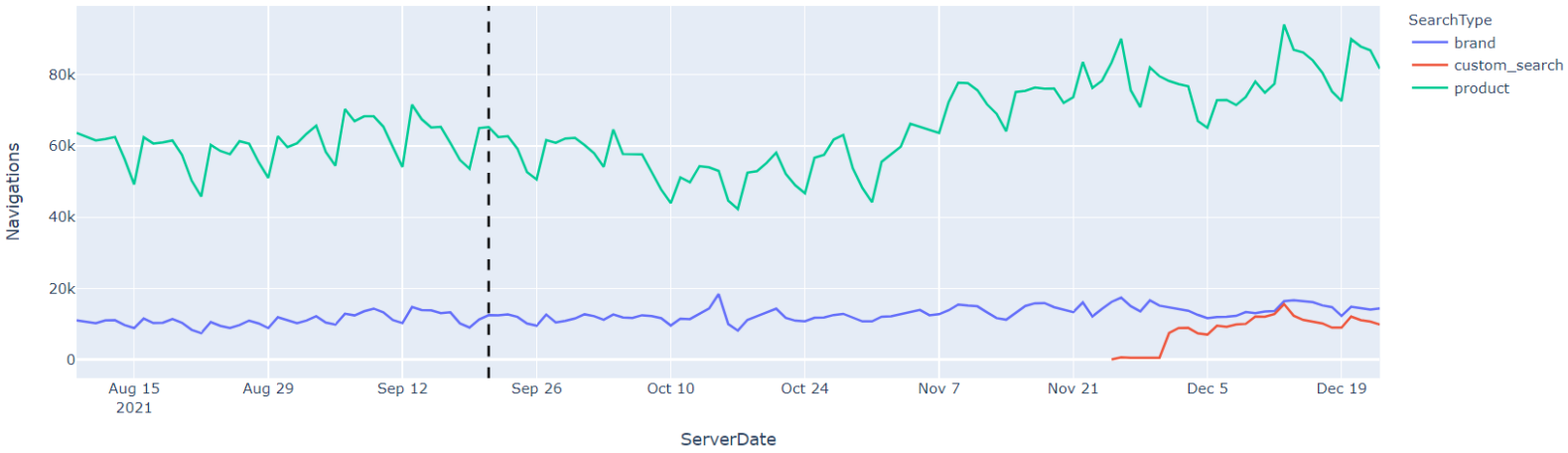


Figura 30: Mostra la distribució de les navegacions, separant per tipus de cerca.
Font: Elaboració pròpia.

Com s'observa, les navegacions de la Collection de marques ha continuat igual (o molt semblant), segurament degut a l'explicat anteriorment, que només hi ha un camp per les marques, mentre que la Collection dels productes, les seves navegacions han crescut de mitjana unes quinze mil - vint mil.
Per últim, cal destacar que a l'inici de les cerques custom, hi havia poques navegacions pel fet que només estava activat a Espanya. A més, aquestes navegacions, que de mitjana es troben al voltant de les deu mil, són navegacions guanyades que abans no es tenien pel fet d'afegir que l'usuari pugui cercar qualsevol cosa. Tot i això, es preveu que aquestes navegacions vagin en augment, ja que si un usuari escriu un text mínimament semblant a una paraula precalculada, aquest text és redirigit a la paraula precalculada, però aquesta funcionalitat es vol modificar en un futur pròxim.

11. Integració de coneixements

Durant el desenvolupament del projecte, s'han posat en pràctica als coneixements adquirits en algunes de les següents assignatures del grau:

- **Bases de dades (BD)**: L'assignatura ha estat útil sobretot per haver après el llenguatge SQL (a l'haver fet queries per retreure informació de la BD) i també entendre més ràpidament el funcionament i relacions entre les taules de la BD.
- **Algorísmia (A) i Estructures de Dades i Algorismes (EDA)**: En tractar-se d'un volum molt elevat de les dades i fer transformacions en aquestes, s'ha de tenir molt en compte el cost temporal i espacial, ja que, o bé pot trigar massa a acabar un algorisme, o bé col·lapsar per no tenir suficient espai en memòria. A més, també s'han de mirar bé les estructures de dades que s'escullen per fer aquest tractament.
- **Llenguatges de Programació (LP)**: A l'aprendre en aquesta assignatura Python i llenguatges funcionals, el fet d'haver de programar en Python ha sigut una tasca més lleugera del que hauria pogut ser.
- **Mineria de Dades (MD)**: En aquesta assignatura, com s'havia d'utilitzar Pandas (una llibreria de Python) i tractar amb un volum mitjà de dades aplicant transformacions, ha fet no arribar amb un coneixement nul en aquest àmbit.
- **Cerca i Anàlisi d'Informació Massiva (CAIM)**: Elasticsearch utilitza moltes tècniques que s'han ensenyat en aquesta assignatura i, per tant, ha sigut molt fàcil comprendre i saber utilitzar-lo havent cursat aquesta assignatura. En aquest projecte, CAIM ha estat l'assignatura més útil i aplicable.
- **Teoria de la Computació (TC)**: En haver de fer un petit parser utilitzant expressions regulars en el projecte, ha fet que sigui tot més senzill havent cursat aquesta assignatura.
- **Projecte Aplicat d'Enginyeria (PAE) i Projectes de Programació (PROP)**: En aquestes assignatures, en haver de treballar en un equip (tot i haver de fer la teva part del treball), fa tenir una certa aproximació al que seria incorporar-se a treballar en un equip desenvolupant software.

12. Lleis i regulacions

Pel que respecte a aquest treball, s'ha de tenir en compte la llei de protecció de dades (LOPD, RGPD, així com les mateixes lleis de cada país que Tiendeo opera i no formen part de la Unió Europea), ja que es té accés a dades pròpies dels usuaris. També s'ha de mirar la regulació de la política de privacitat i cookies, ja que el treball va enfocat a l'usuari.

La *Llei Orgànica 15/1999 de Protecció de Dades de caràcter personal (LOPD)*, com també el *Reglament General de Protecció de Dades (RGPD),* però aquest aplicat a la Unió Europea, esmenta que tota persona física, registrada a un sistema informàtic o no, té dret al seu honor, intimitat, privacitat personal i familiar[31]. Per tant, en el treball s'haurà d'assegurar la seguretat i no expansió de les dades privades dels usuaris, registrats o no a Tiendeo, ja que, encara que aquests no estiguin registrats, es pot saber la seva ubicació en un moment donat, els seus gustos (perquè sabem en tot moment el que ha cercat i el que l'interessa) i més.

Aquesta llei va de la mà amb la política de privacitat i cookies, ja que a l'hora d'entrar a la web, es guarden dades per poder identificar d'una manera o una altra a un individu. Per exemple, depenent de la seva ubicació, a l'hora de cercar els articles, li sortirà uns o altres.
Per tant, com ja es fa a la web, s'ha d'informar a l'usuari de com es tractaran les seves dades i les cookies que es quedaran guardades. D'aquesta forma, es podran oferir resultats d'acord amb les seves característiques.

Per últim, com totes les llibreries i softwares utilitzats són de codi lliure i tenen permès l'ús comercial, no es té en compte cap regulació extra.

13. Conclusions

En aquest apartat, es mirarà de concloure aquest projecte, que, de forma resumida, genera una millora en l'ordenació de productes de les cerques que es troben precalculades i afegeix la funcionalitat de poder cercar qualsevol mena de text que no es troba precalculat, per tant, soluciona completament el problema plantejat inicialment.
Es donarà justificació de l'assoliment dels objectius i les competències tècniques, així com mostrar quins són els propers passos per millorar i continuar desenvolupant el cercador.

13.1. Assoliment dels objectius

Com bé es va mencionar a l'abast del projecte, l'objectiu principal del projecte és **desenvolupar i analitzar un cercador que sigui capaç de fer *full-text search*.** I, en estar ja operatiu en tots els països (la nova ordenació de paraules precalculades i les cerques *full-text*) i haver desenvolupat eines per a poder analitzar i millorar el cercador (Streamlit, com s'ha comentat a la validació dels resultats), s'ha aconseguit assolir l'objectiu principal del projecte.

Pel que respecta els subobjectius del projecte, tenim els següents:

Rendiment.
Com bé s'ha explicat en els apartats Must Query i Definició dels analyzers, s'han adoptat mesures per tal de reduir el temps de cerca d'un usuari. A més, s'ha optat per minimitzar la combinació de queries que conformen la query final, per tal de reduir encara més el temps de cerca.

Investigar i desenvolupar característiques per a poder maximitzar els bons resultats en qualsevol mena de cerca
La combinació de les diferents queries que s'han vist en l'apartat Creació de la query de cerca ha estat la prova i el camí per a poder maximitzar els bons resultats, aprofitant, entre d'altres, la informació que es té de les paraules precalculades per realitzar el càlcul de les seves ordenacions.

Estudi i anàlisi de les interaccions dels usuaris amb el cercador
Com s'ha explicat en la validació dels resultats, s'han anat fent iteracions i desenvolupant eines per tal de veure com interaccionen els usuaris amb el cercador i tractar de millorar-lo.

Anàlisi de la bona configuració del cercador per a cada país

Tal com s'ha comentat en l'apartat de Configuració per a cada idioma, s'ha tractat de fer la millor configuració per a cada idioma, contrastant informació i interactuant amb el feedback implícit que anava donant l'usuari.

Analitzar la necessitat de canviar d'Elasticsearch a una altra tecnologia
En l'apartat Anàlisi de diferents tecnologies, s'ha justificat la no necessitat d'haver de migrar de tecnologia, i també s'ha parlat de si era necessari canviar altres tecnologies.

Desenvolupar un suggeridor pel cercador
S'ha desenvolupat un suggeridor/autocompletat, com bé s'ha definit a l'apartat de Funcionament de l'autocompletat, tot i encara no trobar-se operatiu pels usuaris, aquest ja es troba disponible i a punt per tal que l'ordenació pugui ser consumida per part de l'equip de WEB de Tiendeo.

13.2. Assoliment competències tècniques

En aquesta secció es justifica l'assoliment de les competències tècniques pròpies del projecte.

CCO1.1: Avaluar la complexitat computacional d'un problema, conèixer estratègies algorísmiques que puguin dur a la seva resolució, i recomanar, desenvolupar i implementar la que garanteixi el millor rendiment d'acord amb els requisits establerts.
Com s'ha comentat durant el projecte, s'ha necessitat extreure informació de diferents taules de la BD de Tiendeo, transformar-la i posar-la en camps en específic per a després poder pujar-los a Elasticsearch. Aquesta extracció, i, sobretot, transformació de les dades, ha anat passant per diferents implementacions i millores per tal de garantir la millor complexitat computacional, ja que, a més trigui aquesta transformació d'aquestes dades, més endarrereix processos que s'han d'executar després d'aquest o pot generar que els ítems que es veuen per les cerques en la web de Tiendeo no estiguin actualitzats un dia concret.

CCO1.2: Demostrar coneixement dels fonaments teòrics dels llenguatges de programació i les tècniques de processament lèxic, sintàctic i semàntic associades, i saber aplicar-les per a la creació, el disseny i el processament de llenguatges.
Al tenir fonaments de llenguatges de programació per assignatures cursades durant el grau, com LP, s'ha pogut anar desenvolupant el projecte amb Python, i, en aplicar tècniques de processament del llenguatge natural i llenguatges regulars a textos, s'han acabat aplicant tècniques semblants als tres tipus de processament dels llenguatges de programació.

Per exemple, a Python, hi ha un analitzador lèxic, que desglossa text en tokens[37]. Com bé s'ha explicat a l'apartat Definició dels analyzers, el que s'ha desenvolupat en el cercador és separar les paraules en tokens i analitzar-les de forma individual, idea semblant a la tècnica de processament sintàctic que utilitza Python. A més, a Python, el text és llegit per un parser (un analitzador sintàctic)[37], i això és semblant a algunes regles fetes de llenguatges regulars pel processament de text a l'hora de desenvolupar el cercador.

CCO1.3: Definir, avaluar i seleccionar plataformes de desenvolupament i producció hardware i software per al desenvolupament d'aplicacions i serveis informàtics de diversa complexitat.

Pel que respecta a les plataformes utilitzades pel desenvolupament del cercador (software), es troben especificades en l'apartat Recursos materials. Elasticsearch, Kubernetes i Docker es troben analitzades en la secció Anàlisi de tecnologies, mentre que la resta, han estat seleccionades o bé per ser plataformes ja utilitzades per l'empresa i no hi ha cap potencial de millora en canviar-les (tals com AWS, Jenkins, Gitlab i SourceTree), o bé per ser de lliure ús (Trello) i per la usabilitat que ofereixen (Visual Studio Code, ja que té les característiques necessàries per a desenvolupar el projecte, tals com: Mode debug, afegir les taules de la BD de Tiendeo a l'IDE, etc.).

CCO2.1: Demostrar coneixement dels fonaments, dels paradigmes i de les tècniques pròpies dels sistemes intel·ligents, i analitzar, dissenyar i construir sistemes, serveis i aplicacions informàtiques que utilitzin aquestes tècniques en qualsevol àmbit d'aplicació.

En aquest projecte s'ha tractat de dissenyar i construir un cercador amb les eines que, Elasticsearch, una tecnologia amb tècniques pròpies de sistemes intel·ligents, ofereix.

En utilitzar features tals com el CTR propi d'un ítem, fa que el mateix sistema de recomanació es retro-alimenti de les interaccions dels usuaris i posicioni millor o pitjor alguns ítems. A més, s'ha intentat maximitzar que les cerques que els usuaris cerquen més, siguin, en temps d'execució, baixes. També s'han fet servir mètodes d'NLP al cercador i a l'autocompletat propis de sistemes intel·ligents (com utilitza el cercador de Google). Un exemple explicat en aquest document podria ser l'ús de llenguatges regulars en el processament de dades o tallar algunes paraules mitjançant *stemmers*.

CCO2.4: Demostrar coneixement i desenvolupar tècniques d'aprenentatge computacional; dissenyar i implementar aplicacions i sistemes que les utilitzin, incloent les que es dediquen a l'extracció automàtica d'informació i coneixement a partir de grans volums de dades.

Com s'ha explicat anteriorment, s'han desenvolupat processos de pipelines per tal d'extreure un gran volum d'informació de forma automàtica i posterior transformació.

CCO2.5: Implementar software de cerca d'informació (information retrieval).
El projecte tracta de crear un cercador (que, en si mateix, és un sistema de cerca d'informació) i, per tant, s'ha desenvolupat software, ajudat de la tecnologia Elasticsearch, per tal d'implementar-lo.

CCO3.1: Implementar codi crític seguint criteris de temps d'execució, eficiència i seguretat.
En tot moment, com s'ha explicat prèviament, s'ha tingut en consideració el temps d'execució.
A més, com s'havien de fer transformacions de grans volums de dades, les quals, en executar processos, es deixaven en memòria RAM, s'havia de mesurar molt acuradament la consumició d'aquest recurs, ja que es paral·lelitzen processos en una mateixa màquina. Així com saber mesurar la quantitat de peticions simultànies enviades a les APIs d'Elasticsearch per tal de no saturar els clústers (servidors). Per tant, a l'hora d'implementar el codi per transformar les dades, s'ha tingut l'eficiència en ment.
Pel que respecta la seguretat, en introduir els usuaris textos que, posteriorment, es queden guardats a la BD de Tiendeo, ja es tenia en compte que aquests, no facin SQL injection ni altres tècniques que puguin posar en perill la seguretat de les dades de l'empresa, ja que no s'executa cap text que pugui introduir un usuari.

13.3. Futur del cercador

El cercador que està implementat actualment, és una versió usable i plenament funcional, ja que està actiu a tots els països. Per tant, el futur que té al davant el cercador són petites millores que es poden anar fent tals com incorporar petites característiques, o, fins i tot, canvis en l'estructura de camps en certs països.

Els primers passos per millorar el cercador seria posar en actiu l'autocompletat. D'aquesta manera, quan un usuari volgués buscar algun concepte, primerament sortiria el que més s'està buscant en aquell moment. I, sigui per alguna novetat o interès social, podria fer que es generessin més clics en veure en el desplegable de cerques, un producte del qual l'usuari no era conscient.

També, el marge de millora en els analyzers per cada país és molt gran, ja que les opcions i característiques que es poden aplicar amb NLP són immenses. Per tant, faria millorar encara més el cercador en alguns països.

Pel que respecta als països com el Japó, on la majoria dels ítems són extrets de catàlegs i, per tant, l'únic text que tenen és extret per OCR, fa que la definició d'un ítem sigui baixa i que no hi hagi molts resultats en algunes cerques. Llavors, si s'utilitzessin diferents formes per cercar en aquests tipus de països, podria fer millorar el cercador.

Finalment, com el cercador és adaptable a nova informació que pugui venir per definir millors els ítems i així fer-lo millorar, sempre hi ha un marge de millora si es decideix optar per diferents i nous descriptors pels ítems.

13.4. Conclusió final

De forma resumida, el projecte ha millorat el cercador que hi havia, oferint noves funcionalitats. Aquest, encara té molt marge de millora i es continuarà desenvolupant per fer dels usuaris una millor experiència.

Ha estat un projecte de sis mesos, on s'ha invertit molt de temps en tractar de buscar solucions i noves maneres per tal de millorar les ordenacions, així com tractar d'implementar-lo a Elasticsearch, desenvolupant software al projecte Profiling de Tiendeo.

Per concloure, el cercador ha complert tots els objectius plantejats de forma satisfactòria i, de cara al futur, es plantejaran de nous per fer-lo millorar.

14. Bibliografia

[1] Wikipedia. 2021. Search engine - Wikipedia. [online] Disponible a: https://en.wikipedia.org/wiki/Search_engine

[2] Wikipedia. 2021. Information retrieval - Wikipedia. [online] Disponible a: https://en.wikipedia.org/wiki/Information_retrieval

[3] BroadbandSearch.net. 2021. 27 Mind-blowing Statistics About Google. [online] Disponible a: https://www.broadbandsearch.net/blog/google-statistics-facts

[4] Wikipedia. 2021. tf–idf - Wikipedia. [online] Disponible a: https://en.wikipedia.org/wiki/Tf%E2%80%93idf

[5] Wikipedia. 2021. Pipeline (computing) - Wikipedia. [online] Disponible a: https://en.wikipedia.org/wiki/Pipeline_(computing)

[6] Trello.com. 2021. Trello. [online] Disponible a: https://trello.com/

[7] GitLab. 2021. Iterate faster, innovate together. [online] Disponible a: https://about.gitlab.com/

[8] Code.visualstudio.com. 2021. Visual Studio Code - Code Editing. Redefined. [online] Disponible a: https://code.visualstudio.com/

[9] Wikipedia. 2021. Web API - Wikipedia. [online] Disponible a: https://en.wikipedia.org/wiki/Web_API

[10] GitLab. 2021. What is a continuous integration server?. [online] Disponible a: https://about.gitlab.com/topics/ci-cd/continuous-integration-server/

[11] Glassdoor. 2021. [online] Disponible a: https://www.glassdoor.es/

[12] Amazon.com. 2021. [online] Disponible a: https://www.amazon.com/-/es/Dell-Latitude-5591-Ordenador-Hexa-Core/dp/B07GBYK4TG

[13] Dell. 2021. Dell Optical Mouse- MS116 (BLACK) | Dell USA. [online] Disponible a: https://www.dell.com/en-us/work/shop/dell-optical-mouse-ms116-black/apd/275-bbcb/pc-accessories

[14] Dell. 2021. Teclado multimedia Dell-KB216 - español (QWERTY) - negro | Dell España. [online] Disponible a:

https://www.dell.com/es-es/shop/teclado-multimedia-dell-kb216-espa%C3%B1ol-qwerty-negro/apd/580-adgs/accesorios-para-pc

[15] Amazon.es. 2021. [online] Disponible a: https://www.amazon.es/Monitor-Samsung-S24A310-Panel-1080p/dp/B08KJB7JKV/ref=sr_1_1?adgrpid=56458922136&dchild=1&gclid=CjwKCAjw2P-KBhByEiwADBYWCndAW5meIS0UALghzEI8rD4cmCiMqRxjEHPYddrOBxzWvN4Xj-rTKRoCCSsQAvD_BwE&hvadid=275356305016&hvdev=c&hvlocphy=1005424&hvnetw=g&hvqmt=e&hvrand=8770861655403707550&hvtargid=kwd-300573076959&hydadcr=29957_1781483&keywords=samsung%2Bmonitor%2B24%2Bpulgadas&qid=1633707734&sr=8-1&th=1

[16] OVHcloud. 2021. Precios. [online] Disponible a: https://www.ovhcloud.com/es-es/public-cloud/prices/

[17] Amazon Web Services, Inc. 2021. Precios Amazon Web Service S3 | Amazon Simple Storage Service. [online] Disponible a: https://aws.amazon.com/es/s3/pricing/

[18] Meet BCN - Coworking Barcelona. 2021. Espacio de coworking en Barcelona - Meet BCN - Coworking Barcelona. [online] Disponible a: https://meetbcn.com/coworking-barcelona/espacio-coworking-barcelona/

[19] Onecowork.com. 2021. Coworking Price Plans & Coworking Membership Plans - OneCoWork. [online] Disponible a: https://onecowork.com/plans

[20] Agenciatributaria.es. 2021. 3.5.4 Tabla de amortización simplificada - Agencia Tributaria. [online] Disponible a: https://www.agenciatributaria.es/AEAT.internet/Inicio/Ayuda/Manuales__Folletos_y_Videos/Manuales_practicos/_Ayuda_Folleto_Actividades_economicas/3__Impuesto_sobre_la_Renta_de_las_Personas_Fisicas/3_5_Estimacion_directa_simplificada/3_5_4__Tabla_de_amortizacion_simplificada/3_5_4__Tabla_de_amortizacion_simplificada.html

[21] IKEA. 2021. MICKE Escritorio, blanco 105x50 cm - IKEA. [online] Disponible a: https://www.ikea.com/es/es/p/micke-escritorio-blanco-80213074/

[22] IKEA. 2021. RENBERGET Silla giratoria, Bomstad negro - IKEA. [online] Disponible a: https://www.ikea.com/es/es/p/renberget-silla-giratoria-bomstad-negro-60493546/

[23] Tmb.cat. 2021. Precio tarjeta T-jove metro bus Barcelona | Transports Metropolitans de Barcelona. [online] Disponible a: https://www.tmb.cat/es/tarifas-metro-bus-barcelona/sencillos-e-integrados/t-jove

[24] Tarifaluzhora.es, 2021. Precio de la tarifa de luz por horas. [online] Disponible a: https://tarifaluzhora.es/

[25] OVHcloud. 2021. Manifesto. [online] Disponible a: https://www.ovhcloud.com/en-ie/lp/manifesto/

[26] Vigunu.com. 2021. Cuanta energía consume un servidor y cuanto CO2 emite. [online] Disponible a: https://vigunu.com/eventsnews/2020/10/03/cuanta-energia-consume-un-servidor-y-cuanto-co2-emite/

[27] Cambio climático. 2021. Factor de emisión de la energía eléctrica: el mix eléctrico. [online] Disponible a: https://canviclimatic.gencat.cat/es/actua/factors_demissio_associats_a_lenergia/#:~:text=El%20mix%20de%20la%20red%20el%C3%A9ctrica%20espa%C3%B1ola%20publicado%20por%20la,25%20kg%20CO2%2FkWh.

[28] Elastic.co. 2021. Tune for search speed | Elasticsearch Guide [master] | Elastic. [online] Disponible a: https://www.elastic.co/guide/en/elasticsearch/reference/master/tune-for-search-speed.html

[29] Es.wikipedia.org. 2021. Distancia de Levenshtein - Wikipedia, la enciclopedia libre. [online] Disponible a: https://es.wikipedia.org/wiki/Distancia_de_Levenshtein

[30] En.wikipedia.org. 2021. Discounted cumulative gain - Wikipedia. [online] Disponible a: https://en.wikipedia.org/wiki/Discounted_cumulative_gain

[31] Wikipedia 2021. Ley Orgánica de Protección de Datos de Carácter Personal (España) - Wikipedia. [online] Disponible a: https://es.wikipedia.org/wiki/Ley_Orgánica_de_Protección_de_Datos_de_Carácter_Personal_(España)

[32] Flatiron School 2021. The 10 Best Data Science Programming Languages to Learn in 2021. [online] Disponible a: https://flatironschool.com/blog/data-science-programming-languages

[33] Nubersia 2021. Kubernetes vs Docker Swarm! cuál es la mejor opción?. [online] Disponible a: https://www.nubersia.com/es/blog/kubernetes-vs-docker-swarm/

[34] Softwaretestinghelp.com. 2021. Top 10 Best Container Software in 2021. [online] Disponible a: https://www.softwaretestinghelp.com/container-software/

[35] Es.wikipedia.org. 2021. Reconocimiento óptico de caracteres - Wikipedia, la enciclopedia libre. [online] Disponible a: https://es.wikipedia.org/wiki/Reconocimiento_%C3%B3ptico_de_caracteres

[36] Es.wikipedia.org. 2021. Depuración de programas - Wikipedia, la enciclopedia libre. [online] Disponible a: https://es.wikipedia.org/wiki/Depuraci%C3%B3n_de_programas

[37] Docs.python.org. 2021. 2. Análisis léxico — documentación de Python - 3.10.1. [online] Disponible a: https://docs.python.org/es/3/reference/lexical_analysis.html