



Escuela
Politécnica
Superior

Título del Trabajo

Fin de Grado/Master



Grado en Ingeniería Multimedia

Trabajo Fin de Grado

Autor:

Nombre y Apellidos

Tutor/es:

Nombre y Apellidos

Marzo 2014



Universitat d'Alacant
Universidad de Alicante

Contrast

Subtítulo del proyecto

Autor

Adrián Tomás Vañó

Directores

Gustavo Candela Romero

Departamento de Lenguajes y Sistemas Informáticos

María Dolores Sáez Fernández

Departamento de Lenguajes y Sistemas Informáticos



GRADO EN INGENIERÍA MULTIMEDIA



Escuela
Politécnica
Superior



Universitat d'Alacant
Universidad de Alicante

ALICANTE, 27 de marzo de 2019

Resumen

Se debe incluir en la memoria un resumen de máximo 500 palabras, mejor se hace al final del trabajo con la visión global del mismo. Es obligatorio presentarlo para el tribunal. Se entrega en un fichero a parte y también lo podemos incluir en la memoria. Puede estar en las tres lenguas cas val y en

Preámbulo

Poner aquí un texto breve que debe incluir entre otras:

“las razones que han llevado a la realización del estudio, el tema, la finalidad y el alcance y también los agradecimientos por las ayudas, por ejemplo apoyo económico (becas y subvenciones) y las consultas y discusiones con los tutores y colegas de trabajo. [AENOR, 1997]”

*A mi esposa Marganit, y a mis hijos Ella Rose y Daniel Adams,
sin los cuales habría podido acabar este libro dos años antes*¹

¹Dedicatoria de Joseph J. Roman en “An Introduction to Algebraic Topology”

*Si consigo ver más lejos
es porque he conseguido auparme
a hombros de gigantes*

Isaac Newton.

Índice general

Resumen	v
1. Introducción	1
2. Modelo de negocio	3
2.1. Lean Canvas	3
3. Objetivos	5
3.1. Generales	5
3.2. Específicos	5
4. Marco Teórico	7
4.1. Estado del mercado	7
4.1.1. Agregadores RSS	7
4.1.2. Servicios de seguimiento de prensa digital	8
4.2. Marco legal	8
4.2.1. Reglamento General de Protección de Datos	8
4.2.2. Explotación y uso de contenido de medios digitales	8
4.2.3. Conflicto AEDE con Google Noticias	8
4.3. Estudio de tecnologías para el desarrollo	9
4.3.1. Lenguajes de desarrollo	9
4.3.2. Back-end	10
4.3.3. Front-end	14
5. Metodología	17
5.1. Hitos e iteraciones	17
5.1.1. Hito 1	17
5.1.2. Hito 2	18
5.1.3. Iteración VII	19
5.1.4. Iteración VIII	19
5.2. Herramientas de organización	19
5.3. Herramientas de desarrollo	20
6. Análisis y especificación	23
6.1. Especificación requisitos funcionales	23
6.2. Casos de uso	23

7. Diseño	25
7.1. Arquitectura seleccionada	25
7.2. Tecnologías	25
7.3. Diagrama de clases	26
7.4. Mockups	26
7.4.1. Hito 1	27
7.4.2. Hito 2	28
8. Implementación	31
9. Pruebas y validación	33
10. Resultados	35
11. Conclusiones	37
11.1. Conclusiones	37
11.2. Líneas de trabajo futuras	37
Bibliografía	39
A. Anexo I. Esquema de base de datos	41
B. Anexo II. Requisitos funcionales	43

Índice de figuras

2.1. Lean Canvas de Contrast	4
4.1. Lenguajes más populares en la encuesta de 2018 de Stack Overflow	9
4.2. Motores de búsqueda más populares en https://db-engines.com/en/ranking/search+engine	11
4.3. Soporte oficial de idiomas en CoreNLP	13
5.1. Espacio de trabajo para Contrast	19
5.2. Tablero para Contrast	20
5.3. Integración IFTT de Todoist con Trello	21
7.1. Stack tecnológico utilizado por Contrast	25
7.2. Prototipo de interfaz para página de inicio y resultado búsqueda.	27
7.3. Prototipo de interfaz para noticia leída en modo leer aparte.	27
7.4. Diseño final de interfaz para página de inicio y resultado búsqueda.	28
7.5. Diseño final de interfaz para noticia leída en modo leer aparte.	29

Índice de tablas

Índice de Listados

1. Introducción

En la introducción es donde se hará énfasis a la importancia de la temática, su vigencia y actualidad; se planteará el problema a investigar, así como el propósito o finalidad de la investigación, la aportación que supone este TFG al dominio/sector que se ha investigado. Debemos explicar el contexto donde se ubicará el trabajo.

No hay una longitud estimada, pero se debe tener en cuenta que es lo primero que lee una persona que debe evaluar el trabajo.

Se pueden establecer secciones si así lo requiere la memoria.

2. Modelo de negocio

2.1. Lean Canvas

Esta sección es opcional y depende de la tipología del trabajo a realizar. El siguiente apartado forma parte del estudio de viabilidad del proyecto, en él se debe reflejar y justificar el porqué surge el trabajo. Es importante pensar y plasmar en el trabajo cómo nuestro proyecto va a aportar valor a los clientes, conocer sus necesidades o problemas y cómo les damos solución y cómo estos clientes van a pagar por ese valor que reciben. Además, es importante conocer el tipo de mercado en el que operamos y qué otros factores influyen en él. Se analizan qué alternativas existen hoy en día en ese sector que solucionen total o parcialmente el problema y que afecten a los cliente o bien se ha detectado que no existe una solución (aunque sea comercial) a dicho problema.

Importante detectar problema o necesidad, a quién afecta, cómo se resuelve y qué se va a aportar, es decir cómo se va a solucionar (TFG).

Se pueden incluir análisis de DAFO y análisis de los riesgos

PROBLEM	SOLUTION	UNIQUE VALUE PROPOSITION	UNFAIR ADVANTAGE	CUSTOMER SEGMENTS
Dificultad para leer noticias de distintos periódicos en un mismo lugar Dificultad para contrastar información entre periódicos El seguimiento de medios para usos no-comerciales es caro	Agregador de prensa digital de diferentes fuentes Visor simultáneo de noticias Seguimiento de medios básico, agrupando noticias según temática	Visor para lectura simultánea de prensa digital	Lectura simultánea de noticias procedentes de distintos periódicos	Lectores de prensa digital Periodistas Instituciones con necesidades de recuperación de información en distintas fuentes
EXISTING ALTERNATIVES	KEY METRICS	HIGH-LEVEL CONCEPT	CHANNELS	EARLY ADOPTERS
Agregadores de medios que permiten la lectura de noticias Para contrastar la información hay que acceder a cada noticia por separado El seguimiento de medios se hace a través de empresas	Búsquedas Accesos a la aplicación	Comparador de opinión en prensa	Redes sociales Product Hunt Boca a boca Universidad	Lectores de prensa digital
COST STRUCTURE		REVENUE STREAMS		
Alojamiento en Heroku Difusión Horas de desarrollo		Opción Premium con adición de fuentes personalizadas, posibilidad de realizar recortes, subrayados, etc Venta de aplicación para uso privado en instituciones		

Figura 2.1.: Lean Canvas de Contrast

3. Objetivos

3.1. Generales

El objetivo general del proyecto es crear un portal de noticias capaz de realizar distintas agrupaciones según su contenido, de tal manera que el usuario pueda contrastar la información.

3.2. Específicos

1. Construir una API que actúe como agregador de RSS.
2. Utilizar un motor de búsqueda para indexar el contenido RSS.
3. Utilizar una herramienta de procesamiento de lenguaje natural que ayude a determinar la similitud entre noticias.
4. Diseñar un visor de noticias que permita la lectura de estas.

4. Marco Teórico

La siguiente sección trata el estado de las tecnologías existentes para abordar la construcción de la aplicación, al mismo tiempo se mencionan las alternativas existentes que ofrecen una servicio similar al que plantea Contrast. Por otra parte, se trata el marco en el que se engloba el contexto de la aplicación.

4.1. Estado del mercado

En este apartado se detallan el conjunto de servicios sobre los que Contrast comparte nicho de mercado. Los servicios citados están disponibles en internet.

4.1.1. Agregadores RSS

Existen una gran cantidad de agregadores RSS con diferentes características y funcionalidades. Algunos muy populares entre los consumidores de estos servicios son Feedly, Flipboard e Inoreader. Dichos servicios permiten la búsqueda de fuentes de contenido para su lectura y la suscripción a temas de interés, que a su vez suscribe a varias fuentes relacionadas con el tema.

- **Feedly**¹ es el lector de contenido más popular actualmente. Cuenta con una interfaz clara y simple y permite consumir contenido en diversos formatos: vídeo, noticias de periódicos, tweets, publicaciones de blogs, alertas Google Keywords y por supuesto, RSS. Posee herramientas de integración con servicios de terceros como Evernote o Trello y permite crear tableros compartidos con otros usuarios.
- **Flipboard**² ha sido el agregador de noticias más utilizado durante mucho tiempo y una red social en sí. Posee integraciones con las principales redes sociales para leer contenido de ellas y compartir noticias en las redes desde la aplicación. Tiene un diseño cuidado que asemeja el uso de un magazín físico en cuanto a estructura de las secciones y el deslizamiento de hojas.
- **Inoreader**³ se define entre los usuarios como la alternativa a los anteriores cuando hay algún aspecto entre ellos que no termina de encajar con el usuario. Este lector de contenido posee herramientas para etiquetar y organizar automáticamente la información entrante en busca de ahorro de tiempo por parte del usuario.

¹<https://feedly.com/>

²<https://flipboard.com/>

³<https://www.inoreader.com/>

4.1.2. Servicios de seguimiento de prensa digital

El seguimiento de medios consiste en la detección de toda información publicada sobre un tema concreto. Estos datos se analizan en pos de categorizarlos y mejorar la distribución de los mismos. En el contexto en el que se encuentra Contrast, se estudian las empresas de seguimiento de prensa digital.

- **Pressclipping⁴:** esta empresa ofrece análisis de medios y evaluación de noticias, además del propio seguimiento de medios. Cuenta además con soluciones para la gestión de las relaciones públicas y del marketing estratégico para medir con precisión el alcance y reputación de una marca.
- **Puntonews⁵:** servicio que permite buscar noticias tanto de periódico, radio como de televisión y exportarlas a PDF.
- **MyNews⁶:** hemeroteca digital desde 1996. Dispone de seguimiento de formatos audiovisuales y redes sociales, al igual que análisis de presencia y reputación de marca.

4.2. Marco legal

4.2.1. Reglamento General de Protección de Datos

4.2.2. Explotación y uso de contenido de medios digitales

4.2.3. Conflicto AEDE con Google Noticias

Google Noticias es un agregador y buscador de noticias de medios de prensa digital que cuenta con más de 70 ediciones en 35 idiomas y que genera más de 10.000 millones de clics al mes. Esta herramienta, nacida en 2002, no está disponible en España desde hace unos años.

El 5 de noviembre de 2014, el Partido Popular aprobó la nueva ley de propiedad intelectual (<https://www.boe.es/boe/dias/2014/11/05/pdfs/BOE-A-2014-11404.pdf>), causando que Google se viese obligado a pagar una tasa exigida por la Asociación de Editores de Diarios Españoles (AEDE). Dicho de otra manera, ahora tenía que pagar por mostrar cualquier fragmento del contenido de los editores españoles.

Cabe destacar que Google no tenía un modelo de negocio con este servicio, pues no mostraba anuncios, tan solo enlazaba los titulares con la noticia en el medio digital.

La ley aprobada entraba en vigor el 1 de enero de 2015 y el 16 de diciembre Google anunciaba que retiraba el servicio de noticias en España. Los editores contrastaron días después que la plataforma de Google les aportaba tráfico y por lo tanto ingresos por anuncios, por ello pidieron al gobierno y a la Unión Europea que interviniieran, abogando

⁴<https://www.pressclipping.com/>

⁵<https://www.puntonews.com/es>

⁶<https://www.mynews.es/>

que el gigante tenía la posibilidad de negociar la tasa que debía pagar. Google fue tajante con su rehuso.

Actualmente la ley sigue en vigor.

4.3. Estudio de tecnologías para el desarrollo

En esta sección se detallan las tecnologías que puede necesitar el proyecto para su desarrollo. Está dividido en tres subsecciones. Por un lado se abarcan los lenguajes de programación, por otro las herramientas pertenecientes al back-end y por último las que atañen al front-end.

4.3.1. Lenguajes de desarrollo

Se han analizado los lenguajes más populares en la última encuesta publicada en Stack Overflow⁷, que permiten tanto el desarrollo front-end como back-end, aunque se elijan opciones distintas para cada apartado.



Figura 4.1.: Lenguajes más populares en la encuesta de 2018 de Stack Overflow

- **Javascript**⁸ fue el primer lenguaje de scripting para web. Es uno de los lenguajes más usados en web tanto para el back-end como para el front-end, integrado en la gran cantidad de frameworks que se han construído sobre él. Con la salida de ECMAScript 6 se han solucionado muchos problemas que acarreaba el uso de Javascript y se han añadido nuevas funcionalidades que lo convierten en aún

⁷<https://insights.stackoverflow.com/survey/2018/>

⁸<https://www.javascript.com/>

más interesante. Tiene una de las bases de usuarios más sólidas entre todos los lenguajes.

- **Java⁹** es un lenguaje con madurez y con muchos proyectos construidos sobre él que dependen de la instalación de Java en la máquina local. Se caracteriza por la solidez, confiabilidad, extensibilidad y rapidez. Existen multitud de frameworks y plugins que operan sobre Java y la documentación y comunidad es más que extensa. A este razonamiento se le suma el hecho de que es el lenguaje para el desarrollo en Android, por lo que es una opción muy interesante para aprender y trabajar en ella.
- **Python¹⁰** es un lenguaje en alza desde los últimos años. Está diseñado partiendo de la intuitividad y del aproximamiento al lenguaje humano. La popularidad del lenguaje se debe al crecimiento del big data y del internet de las cosas, ya que ofrece algunas herramientas para el prototipado y las analíticas. Es un lenguaje ideal para principiantes, principalmente debido a su flexibilidad.

<https://www.quora.com/Which-language-has-the-best-future-prospects-Python-Java-or-JavaScript> <https://stackshare.io/stackups/java-vs-javascript-vs-python>

4.3.2. Back-end

El siguiente apartado contiene un sumario de tecnologías que posibilitan el back-end del proyecto. Por cada una de estas, se han detallado dos o tres propuestas en función de la popularidad, extensión y uso.

Parseador RSS

Dado que el proyecto debe obtener contenido RSS de distintas fuentes, necesita por lo tanto un transformador del contenido en un formato único con el que pueda trabajar el indexador. En este caso, se ha analizado la herramienta más popular para este cometido en cada uno de los lenguajes citados anteriormente.

- **rss-parser¹¹** es una librería mínima para convertir contenido RSS XML en objetos Javascript. Se compone de una función asíncrona que utiliza en un capa inferior un conversor de XML a objeto Javascript llamado xml2js¹².
- **Universal Feed Parser¹³** es un módulo Python para descargar y parsear feeds. Es un plugin simple, autocontenido en un único fichero y con una única función primaria para parsear. Tiene una documentación extensa y funciones adicionales para sanitizar y parsear otros elementos contenidos en el feed. Trata también con el formato Atom.

⁹<https://www.oracle.com/technetwork/java/javase/overview/index.html>

¹⁰<https://www.python.org/>

¹¹<https://www.npmjs.com/package/rss-parser>

¹²<https://www.npmjs.com/package/xml2js>

¹³<https://pythonhosted.org/feedparser/>

- **Rome**¹⁴ es un framework de Java de código abierto para tratar con RSS y Atom. Incluye parseadores, generadores y conversores entre ambos formatos. Los parseadores, el objeto de interés de Contrast, devuelve generalmente un objeto genérico que permite trabajar sin importar el formato de entrada o salida. Está compuesto por distintos módulos según se realice una acción concreta u otra. Posee una documentación sólida y es ampliamente utilizado para este fin.

Motores de búsqueda e indexadores

El contenido obtenido mediante los feeds anteriormente, debe ser almacenado e indexado, de tal manera que la búsqueda y recuperación de estos datos sea rápida y eficiente. En este sentido es de especial interés el uso de un sistema de recuperación de información con índice invertido, caracterizado por ser una estructura de datos capaz de manejar grandes volúmenes de información orientados a texto. En esta categoría se han investigado los tres motores de búsqueda más populares. <http://www.sisorbe.com/ExamenFinalRI/invertidos.html>

18 systems in ranking, December 2018										
Rank			DBMS	Database Model	Score			Dec 2018	Nov 2018	Dec 2017
Dec 2018	Nov 2018	Dec 2017			Dec	Nov	Dec			
1.	1.	1.	Elasticsearch	Search engine	144.70	+1.24	+24.92			
2.	2.	↑ 3.	Splunk	Search engine	82.18	+1.81	+18.39			
3.	3.	↓ 2.	Solr	Search engine	61.35	+0.47	-4.95			
4.	4.	4.	MarkLogic	Multi-model	14.28	+0.81	+3.13			
5.	5.	5.	Sphinx	Search engine	7.82	+0.46	+1.79			
6.	6.	6.	Microsoft Azure Search	Search engine	5.67	+0.36	+1.56			
7.	7.	7.	Algolia	Search engine	3.62	-0.13	+0.57			
8.	8.	↑ 9.	Google Search Appliance	Search engine	2.93	+0.03	+0.20			
9.	9.	↓ 8.	Amazon CloudSearch	Search engine	2.64	-0.11	-0.09			
10.	10.	10.	Xapian	Search engine	0.60	-0.08	-0.02			
11.	11.	11.	CrateDB	Multi-model	0.44	-0.06	-0.15			
12.	12.	12.	SearchBlox	Search engine	0.24	0.00	+0.01			
13.	13.	↑ 14.	searchxml	Multi-model	0.09	-0.01	+0.09			
14.	14.	↓ 13.	DBSight	Search engine	0.06	+0.00	+0.05			
15.	15.	↓ 14.	Manticore Search	Search engine	0.04	-0.01	+0.04			
16.			FinchDB	Multi-model	0.03					
17.	↓ 16.	↓ 14.	Exorbyte	Search engine	0.01	+0.01	+0.01			
18.	↓ 16.	↓ 14.	Indica	Search engine	0.00	±0.00	±0.00			

Figura 4.2.: Motores de búsqueda más populares en <https://db-engines.com/en/ranking/search+engine>

Apache Solr¹⁵ y **Elasticsearch**¹⁶ están construidas sobre **Apache Lucene**¹⁷. **Lucene** es una potente librería de recuperación de información. Se encarga de construir el índice invertido mencionado anteriormente, una estructura especializada en emparejar

¹⁴<https://rometools.github.io/rome/>

¹⁵<https://lucene.apache.org/solr/>

¹⁶<https://www.elastic.co/es/>

¹⁷<https://lucene.apache.org/>

documentos de texto con términos de consulta. **Lucene**, al igual que otros motores de esta clase, se distingue por la escalabilidad, el rápido despliegue, el manejo de grandes volúmenes de datos, la optimización orientada a búsqueda y esta a su vez, en textos.

Solr es un servidor construido con la librería de **Lucene**. Consiste en una interfaz web de administración, APIs en distintos lenguajes para la realización de consultas y una serie de mejoras funcionales sobre las características base de **Lucene**.

Elasticsearch ha basado su modelo en una API REST Web al ser un motor de búsqueda más reciente y por ello ha ganado popularidad desde la última mitad de década. No existen grandes diferencias entre esta herramienta y la anterior, pero algunas destacables son:

- **Solr** posee una documentación más extensa y de mayor calidad al ser open source y llevar muchos años disponible.
- **Elasticsearch** tiende a tener una mejor ejecución en las consultas analíticas y **Solr** en las orientadas a texto.
- **Elasticsearch** es más intuitivo que **Solr**, especialmente para desarrolladores que se inician en este tipo de tecnologías.
- **Elasticsearch** tiene una escalabilidad inherente más fácil de conseguir que **Solr**, que se desarrolla mediante **SolrCloud**¹⁸.

Por otra parte **Splunk**¹⁹ es un motor de búsqueda y plataforma orientado a las analíticas para el Big Data, es por ello que se ha desestimado su análisis.

Herramientas de procesamiento de lenguajes naturales

Para determinar que un grupo de noticias son similares en distintos periódicos, no solo basta con realizar una búsqueda por tema, también es necesario comprobar que las noticias tengan palabras clave similares. Es por ello que Contrast necesita incorporar una herramienta de procesamiento de lenguaje natural. Este tipo de herramientas realizan análisis morfológicos, sintácticos, semánticos y pragmáticos. En el caso de Contrast, los más interesantes son los análisis NER (Named Entity Recognition), dado que el interés recae en conocer las palabras clave del texto. Existen muchas propuestas en este ámbito, entre ellas destacan: Stanford CoreNLP, Apache OpenNLP y Freeling.

- **CoreNLP**²⁰ es un proyecto de la Universidad de Stanford que provee una serie de herramientas de análisis de lenguaje humano. Proporciona instrumentos para un amplio análisis gramático, además de extractores muy diversos. Entre estos se halla uno de especial interés para el proyecto, la herramienta de reconocimiento de entidades (NER). CoreNLP posee una documentación extensa ya que es ampliamente utilizado, complementándose además con actualizaciones frecuentes que mejoran

¹⁸<https://lucene.apache.org/solr/guide/6.6/solrcloud.html>

¹⁹<https://www.splunk.com/>

²⁰<https://stanfordnlp.github.io/CoreNLP/>

el producto. Posee soporte oficial para seis idiomas (entre ellos el español), aunque no están disponibles todas las funciones, como se puede observar en la figura 4.3 de la página 13. La integración con otros proyectos se realiza a través de una serie de APIs en distintos lenguajes. Esta herramienta es utilizada por la Biblioteca Virtual Miguel de Cervantes²¹, concretamente el analizador sintáctico.

ANNOTATOR	AR	ZH	EN	FR	DE	ES
Tokenize / Segment	✓	✓	✓	✓		✓
Sentence Split	✓	✓	✓	✓	✓	✓
Part of Speech	✓	✓	✓	✓	✓	✓
Lemma				✓		
Named Entities			✓	✓		✓
Constituency Parsing	✓	✓	✓	✓	✓	✓
Dependency Parsing		✓	✓	✓	✓	
Sentiment Analysis				✓		
Mention Detection		✓	✓			
Coreference		✓	✓			
Open IE				✓		

Figura 4.3.: Soporte oficial de idiomas en CoreNLP

- **OpenNLP**²² es un proyecto Apache, consistente en una librería de aprendizaje automático cuyo objetivo es el procesamiento del lenguaje humano en texto. Posee soporte para las tareas NLP más comunes. No tiene soporte al uso de distintos lenguajes, pero posee una serie de modelos entrenados en diferentes idiomas para propósitos determinados. En el caso del español encontramos cuatro modelos entrenados en reconocimiento de entidades: personas, organizaciones, localizaciones y misceláneo. Cualquier extractor de interés que no esté en los modelos tendría que ser entrenado para su propósito.
- **Freeling**²³ es un proyecto de la Universitat Politènica de Catalunya. Se trata de

²¹<http://data.cervantesvirtual.com/analizador-sintactico-automatico>

²²<https://opennlp.apache.org/>

²³<http://nlp.lsi.upc.edu/freeling/node/1>

una librería con una serie de herramientas open source para el análisis de lenguaje humano. Está escrita en C++ y posee una extensa variedad de utilidades, entre ellas las de interés para el proyecto. Tiene soporte para muchos idiomas y dialectos españoles, pero carece de una comunidad sólida que utilice la herramienta de manera asidua, al igual que una documentación no tan extensa como las propuestas anteriores.

Frameworks de desarrollo

Para esta sección se ha detallado un framework por cada lenguaje que se ha analizado previamente. Esta selección de frameworks ha sido realizada bajo criterios de popularidad, mantenimiento, calidad en la documentación, extensibilidad e integración con el resto de herramientas que componen el entorno de Contrast.

- **Spring Boot**²⁴ es un framework basado en Spring para Java. Abstacta el proceso de elección de dependencias y despliegue del proyecto, realizando en cambio un proceso sencillo y automatizado. El objetivo principal de esta herramienta es proporcionar un conjunto de herramientas para construir rápidamente aplicaciones en el potente entorno Spring. Se trata de un framework maduro, con resultados profesionales e integraciones con otros productos de Spring que añaden más funcionalidades. Cabe destacar que posee una capa de abstracción²⁵ para el motor de búsqueda Solr, al igual que otra para Elasticsearch²⁶, aunque la primera está más consolidada.
- **Express.js**²⁷ es un framework rápido, minimalista y flexible para el entorno de ejecución Node.js para Javascript. Proporciona un conjunto sólido de características para realizar APIs a través de métodos y utilidades HTTP y middleware que garantizan una creación rápida y sencilla.
- **Django**²⁸ es un framework full-stack para Python que facilita rapidez y claridad, ya que se hace cargo de muchos conceptos que conlleva el desarrollo web. Es una herramienta altamente escalable debido a su flexibilidad y proporciona ayuda a los desarrolladores para evitar muchos errores comunes de seguridad.

4.3.3. Front-end

El siguiente apartado contiene un sumario de tecnologías que posibilitan el front-end y diseño de interfaces del proyecto. Por cada una de estas, se han detallado dos o tres propuestas en función de la popularidad, extensión y uso.

²⁴<https://spring.io/>

²⁵<https://spring.io/projects/spring-data-solr>

²⁶<https://spring.io/projects/spring-data-elasticsearch>

²⁷<https://expressjs.com/>

²⁸<https://www.djangoproject.com/>

Frameworks de desarrollo

Dentro de los frameworks para el desarrollo del front-end, se puede escoger una herramienta de un lenguaje distinto al del back-end. En este caso, se detalla un conjunto de frameworks con gran popularidad en la actualidad. Todos ellos basados en Javascript, el lenguaje de front-end que más alcance tiene estos días.

- **Angular²⁹** es una plataforma para diseñar aplicaciones multidispositivo creada por Google que facilita el desarrollo de aplicaciones web SPA, proporcionando herramientas para trabajar con los elementos de la página web de una manera óptima. Tiene una gran comunidad detrás y una documentación extensa, pero en cierto modo fuerza al usuario a aprender TypeScript.
- **Vue³⁰** es un framework progresivo con un ecosistema que se plantea cercano, versátil y con gran rendimiento. Esta herramienta, diseñada para construir interfaces de usuario reactivas, se creó con el propósito de ser fácil de integrar con otras librerías o proyectos a través del desacople de las partes que lo conforman. Es un framework relativamente nuevo que mejora algunos conceptos de Angular y React, a través de una documentación clara y detallada.
- **React³¹** es una librería para construir interfaces de usuario reactivas SPA. Esta herramienta de Facebook permite la creación de aplicaciones web potentes que requieran un intercambio de datos constante. Tiene una curva de aprendizaje algo elevada que recompensa con eficiencia y responsividad.

Librerías de diseño

En este apartado se detallan tres librerías de diseño de interfaces, que complementan el front-end con animaciones y elementos preconstruídos. La selección ha sido realizada en base al número de estrellas que acumula cada proyecto en Github.

- **Bootstrap³²** es la librería de diseño más popular para construir aplicaciones web responsive pensadas en el mobile first. Dispone elementos que al utilizarlos aplican diferentes propiedades HTML, CSS y JS con un esfuerzo mínimo. Posee una gran comunidad detrás que además contribuye con plugins extra.
- **Foundation³³** es una librería open-source que incluye semántica. Su uso es intuitivo, complementándose con entrenamiento, soporte y consultas, por lo tanto garantiza una gran ayuda para los desarrolladores. Tiene además la capacidad de ser utilizado para diseñar plantillas de correo electrónico.

²⁹<https://angular.io/>

³⁰<https://vuejs.org/>

³¹<https://reactjs.org/>

³²<https://getbootstrap.com/>

³³<https://foundation.zurb.com/>

- **Semantic UI**³⁴ posee el enfoque en realizar más semántico el proceso de construcción de una página. La principal funcionalidad consiste en utilizar principios del lenguaje natural para hacer el código más legible y fácil de entender. Posee una documentación bien organizada y con múltiples guías para empezar.

³⁴<https://semantic-ui.com/>

5. Metodología

El proyecto ha seguido un modelo de desarrollo incremental mediante dos hitos compuestos por iteraciones cada dos semanas. Al finalizar cada iteración se ha realizado una revisión con los tutores del proyecto con el fin de analizar el avance, responder dudas surgidas durante el sprint y planificar el siguiente. Al finalizar el primer hito, con un desarrollo funcional de la mayoría de características, se empezó el segundo desde el principio evitando los errores realizados en el primer hito y mejorando la calidad del código. El trabajo realizado en cada hito e iteración se detalla en el siguiente apartado.

5.1. Hitos e iteraciones

Este apartado trata la duración y tareas propuestas en cada iteración. Se especifican las tareas a grandes rasgos, sin entrar en detalles o subtareas.

5.1.1. Hito 1

El primer hito ha consistido en realizar una primera implementación del proyecto, probando las diferentes propuestas tecnológicas y analizando las interacciones e integraciones entre ellas.

Iteración I

La primera iteración ha analizado el problema a tratar así como las tecnologías para abordarlo y qué se quiere mostrar. Concretamente las tareas han sido:

- Estudiar el marco teórico y documentarlo.
- Establecer metodología de trabajo.
- Establecer objetivos.
- Describir requisitos funcionales.

Iteración II

La segunda iteración ha consistido en especificar ciertas bases del proyecto y construir una implementación inicial de algunas partes independientes probando diferentes tecnologías. Las tareas han sido:

- Prototipar de interfaces.

- Probar y escoger tecnologías para el desarrollo y documentarlo.
- Construir parseador RSS.
- Definir casos de uso de la aplicación.

Iteración III

La tercera iteración ha pretendido conseguir una API REST funcional mínima que interactuase con el motor de base de datos. Las tareas han sido:

- Definir modelo de negocio.
- Construir API REST con CRUD mínimo.
- Definir esquema de datos provisional e integrar motor de búsqueda.

Iteración IV

En la cuarta iteración se ha integrado un framework de front-end que muestre los resultados y la herramienta de procesamiento de lenguaje natural. El objetivo era llegar a una primera versión semi-funcional.

- Construir front-end mínimo y funcional e integrarlo con la API REST para mostrar los resultados.
- Documentar implementación inicial del proyecto.
- Integrar herramienta de procesamiento de lenguaje natural.

5.1.2. Hito 2

El segundo hito ha tomado como referencia el resultado implementado en el primero, ha replanteado algunas de sus elecciones del stack tecnológico y ha partido de una base más sólida sobre la que desarrollar.

Iteración V

- Estudio e integración de tecnologías CI/CD y contenedores.
- Documentar arquitectura seleccionada.
- Documentar diseño final de la aplicación.

Iteración VI

- Detallar esquema de datos definitivo y optimizar la indexación.
- Construir facetados y aspectos avanzados de la API.

5.1.3. Iteración VII

- Optimizar el uso de la herramienta de NLP para disminuir el coste computacional.
- Reconstruir el front-end para mostrar el diseño final.

5.1.4. Iteración VIII

-
-

5.2. Herramientas de organización

A continuación se detallan las herramientas que se han empleado para la organización del proyecto:

- **Workona¹:** extensión del navegador Chrome que facilita la gestión de las pestañas. Su principal funcionalidad es crear espacios de trabajo en el que se guardan las pestañas actualmente abiertas de una sesión a otra. Esto permite cambiar el dispositivo de trabajo fácilmente sin perder el contenido que se estaba visitando en ese momento. Al mismo tiempo también permite cambiar el proyecto en el que se esté trabajando con un solo clic.

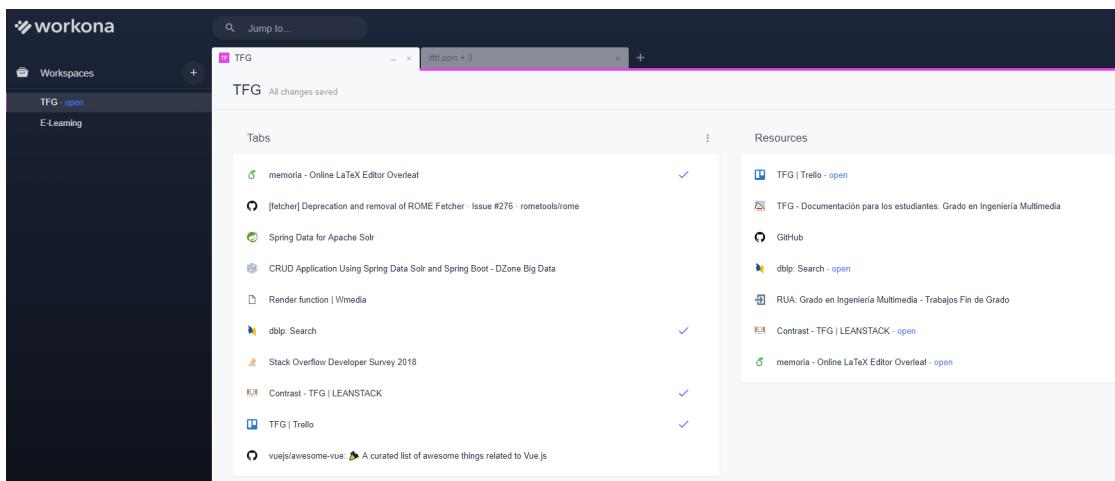


Figura 5.1.: Espacio de trabajo para Contrast

- **Trello²:** aplicación web que posibilita la creación de tableros con listas de tareas. Dichas tareas pueden ser etiquetadas y comentadas y se les pueden añadir archivos adjuntos. Los tableros permiten el flujo libre de tareas entre una lista y otra.

¹<https://workona.com/>

²<https://trello.com/>

En el caso del proyecto, se ha utilizado una metodología de gestión de trabajo Kanban, por lo tanto se han creado tres listas: por hacer, en proceso y finalizado. Se han añadido dos listas adicionales: una para acumular las tareas al finalizar una iteración y otra para definir tareas que surgen durante el proceso de desarrollo y así asignarlas en una iteración próxima.

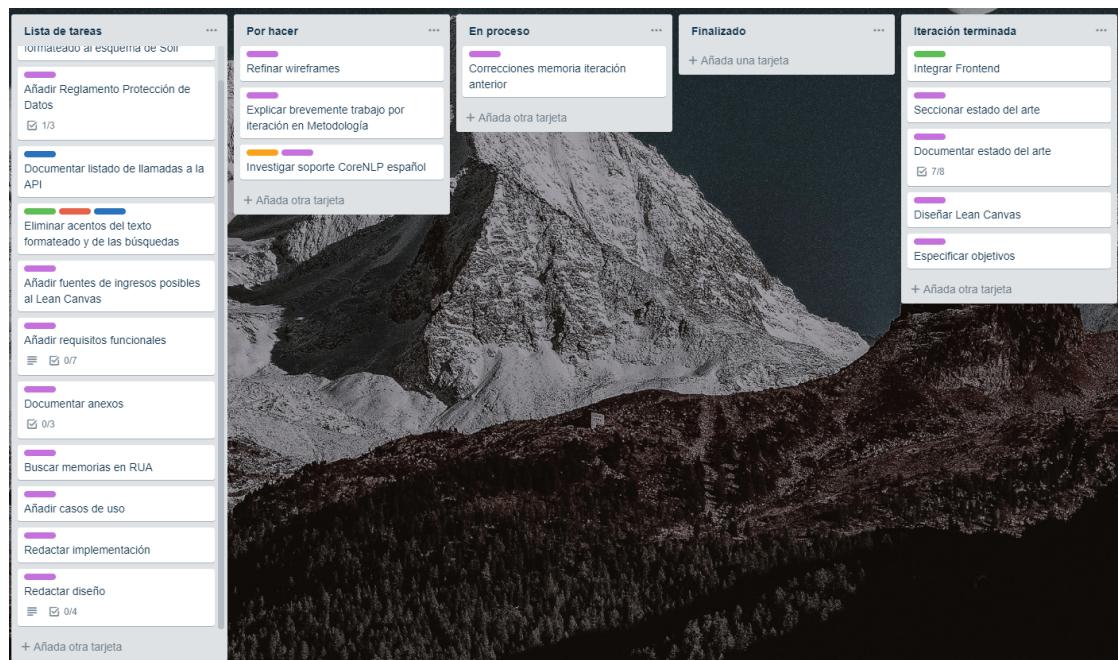


Figura 5.2.: Tablero para Contrast

- **Todoist³:** aplicación web que facilita la organización de tareas en el día a día. En el caso del proyecto es utilizada para organizar las tareas personales junto a las del trabajo de final de grado. Esta aplicación es utilizada junto a IFTTT⁴ (If This Then That), que posibilita la integración de Todoist y Trello, de tal manera que cada vez que se cree una tarjeta en Trello, se añade una tarea en Todoist. La aplicación es utilizada bajo la filosofía GTD⁵ (Getting Things Done).

5.3. Herramientas de desarrollo

En esta sección se especifican las herramientas utilizadas para el desarrollo del proyecto:

- **Visual Studio Code⁶:** editor de código fuente multiplataforma, open-source que

³<https://todoist.com/>

⁴<https://ifttt.com/>

⁵https://es.wikipedia.org/wiki/Getting_Things_Done

⁶<https://code.visualstudio.com/>



Figura 5.3.: Integración IFTTT de Todoist con Trello

incluye muchas funcionalidades en forma de extensiones para trabajar con todos y cada uno de los lenguajes existentes. Su característica principal es el IntelliSense, que autocompleta las instrucciones mientras se escribe y destaca elementos con los que se interactuan.

- **GitKraken⁷:** cliente de Git⁸ con una interfaz visual que automatiza ciertos procesos internos de las instrucciones y permite un uso intuitivo de Git. Destaca por el uso de la técnica de *arrastrar y soltar* para interactuar con los *commits*.
- **Postman⁹:** herramienta que facilita el desarrollo, prueba y validación de elementos en una API REST. Nació como una extensión de Chrome que evolucionó en multiplataforma; hoy en día se ha convertido en un *must-have* del desarrollo web.
- **Overleaf¹⁰:** servicio de LaTeX¹¹ colaborativo en línea con el que redactar y publicar documentos académicos de una manera más rápida, visualizando el resultado tras cada modificación automáticamente.

⁷<https://www.gitkraken.com/>

⁸<https://git-scm.com/>

⁹<https://www.getpostman.com/>

¹⁰<https://www.overleaf.com/>

¹¹<https://www.latex-project.org/>

6. Análisis y especificación

6.1. Especificación requisitos funcionales

En el Anexo B se detallan los requisitos funcionales de la aplicación.

6.2. Casos de uso

7. Diseño

7.1. Arquitectura seleccionada

En la figura 7.1 se detalla el conjunto de tecnologías seleccionadas para el desarrollo del proyecto. Cabe destacar que algunas se han omitido, ya que sirven de soporte a otras que sí que se han detallado.

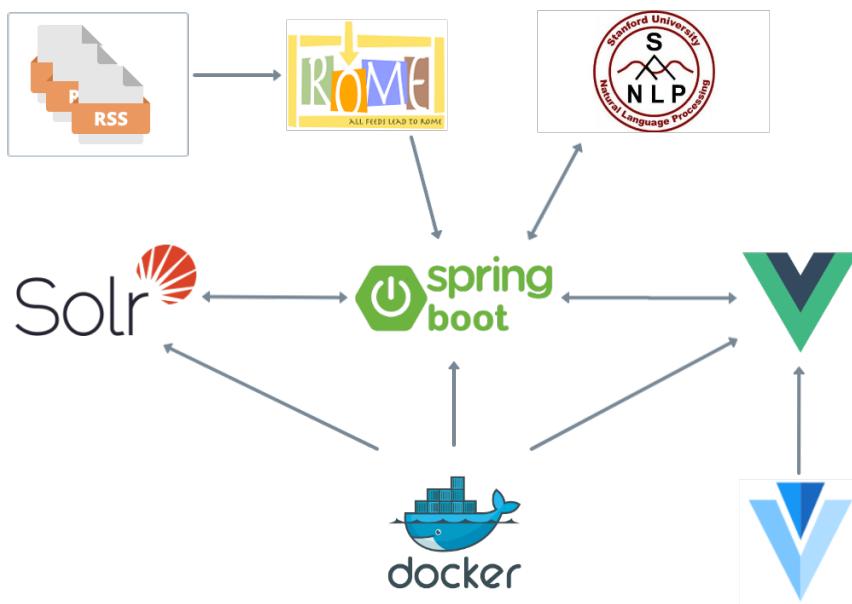


Figura 7.1.: Stack tecnológico utilizado por Contrast.

7.2. Tecnologías

El siguiente apartado justifica las tecnologías escogidas entre las analizadas en la sección 4, Marco Teórico. Por una parte, se ha optado por utilizar Java y por lo tanto Spring Boot, en el backend por la gran compatibilidad que ofrece con el resto de tecnologías que se tenían que integrar con él, por su madurez y por su extensa documentación. Esta decisión también se ha visto condicionada por:

- Tener una capa de abstracción para tratar directamente con el motor de elección de base de datos.

- Tener la posibilidad de utilizar CoreNLP mediante una dependencia del proyecto, simplificando su uso e integración.

Dado que se ha escogido Java, la elección de Rome como parseador era muy clara. Esto es debido a que se integra del mismo modo como una dependencia más del proyecto, con el añadido de ser además la herramienta más potente de las tres analizadas.

Entre los motores de búsqueda la elección ha sido más difusa, de hecho se ha realizado el desarrollo prácticamente paralelamente sobre ambos. Esto ha sido posible gracias a que la capa de abstracción de Spring Boot sobre Apache Solr y Elasticsearch, permitía con modificaciones mínimas cambiar entre un motor u otro. Finalmente se ha escogido Solr dado que posibilitaba crear previamente el esquema de datos mediante una sencilla declaración de parámetros en un XML.

Como herramienta de procesamiento de lenguaje natural, la balanza se ha inclinado hacia CoreNLP por las razones que se detallan a continuación:

- CoreNLP se ha integrado con el backend escogido como una dependencia más, simplificando su uso.
- CoreNLP posee una mayor documentación que el resto analizado, facilitando el entendimiento del uso de los extractores.
- CoreNLP es la herramienta más utilizada de las tres analizadas en este ámbito.

Por último, el frontend se ha construido con Vue.js. Dado que la aplicación requiere de pocas interfaces y pocos elementos en esta, este framework posibilita un rápido desarrollo con componentes preconstruidos sobre los que tan solo había que enlazar los datos. Encima de este se ha utilizado Vuetify como librería de diseño y componentes, que al ser nativa de Vue, ha permitido embellecer el resultado sin realizar excesivas modificaciones.

7.3. Diagrama de clases

7.4. Mockups

En la siguiente sección se detallan los mockups que se realizaron para ambos Hitos. El primero plasma un esbozo de la idea original, sin entrar en detalles de su posterior implementación y el segundo, en cambio, muestra la idea refinada con un aspecto similar al de su implementación final.

7.4.1. Hito 1

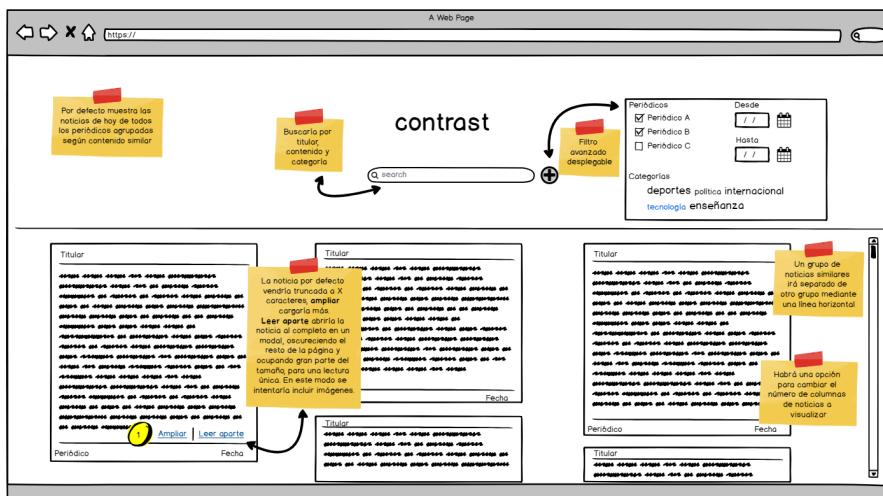


Figura 7.2.: Prototipo de interfaz para página de inicio y resultado búsqueda.

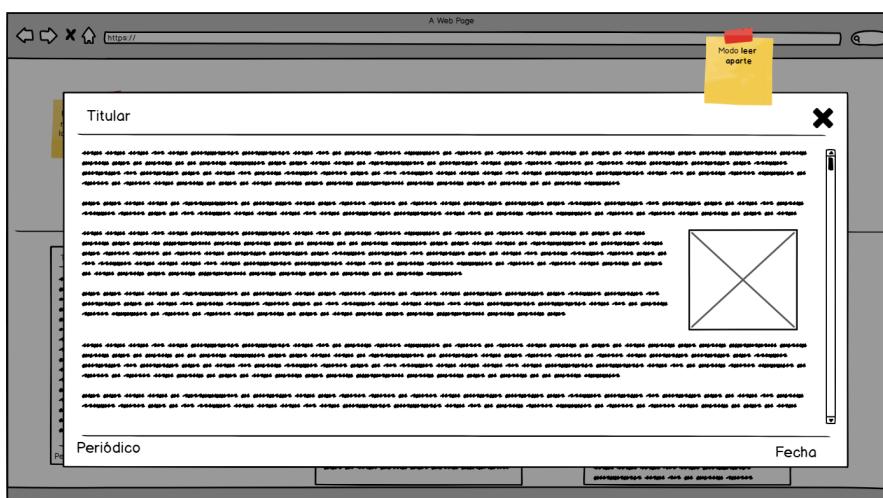


Figura 7.3.: Prototipo de interfaz para noticia leída en modo leer aparte.

7.4.2. Hito 2

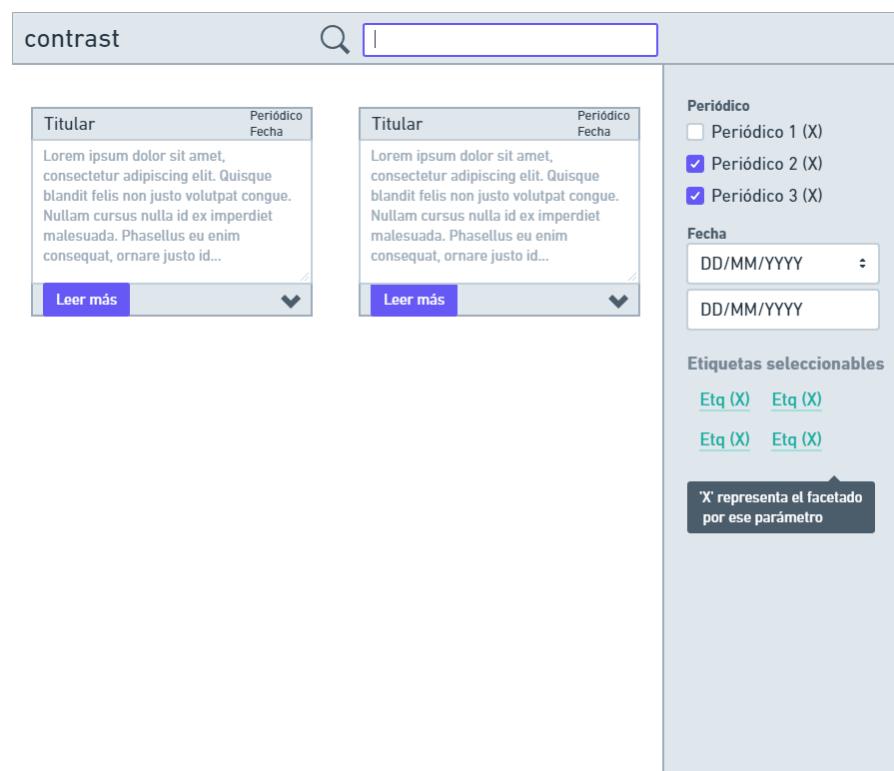


Figura 7.4.: Diseño final de interfaz para página de inicio y resultado búsqueda.

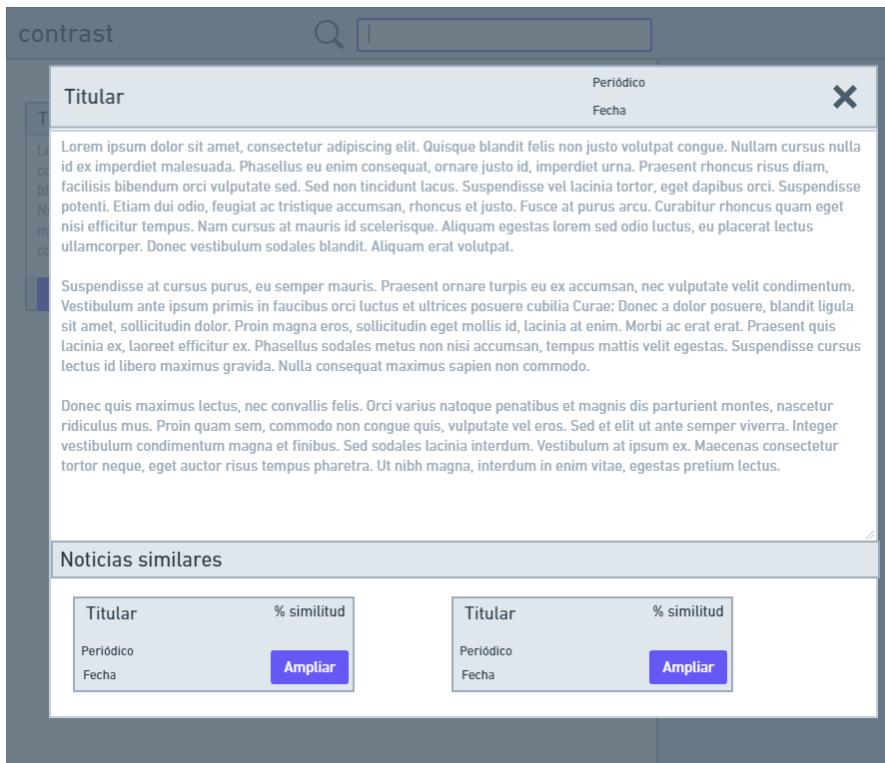


Figura 7.5.: Diseño final de interfaz para noticia leída en modo leer aparte.

8. Implementación

9. Pruebas y validación

10. Resultados

11. Conclusiones

11.1. Conclusiones

11.2. Líneas de trabajo futuras

Bibliografía

[AENOR, 1997] AENOR (1997). norma une 50136:1997.

A. Anexo I. Esquema de base de datos

Aquí vendría el anexo I

B. Anexo II. Requisitos funcionales

Identificador del requisito	RF01
Fuente del requisito	Aplicación web
Nombre	Listado de noticias
Descripción	La aplicación mostrará un listado de noticias recientes.
Características	<ul style="list-style-type: none">■ Las noticias estarán agrupadas según la relación entre ellas.■ Cada noticia de la lista contendrá, al menos, el titular, la fecha de publicación, el enlace y el periódico al que pertenece.■ Las noticias vendrán truncadas por defecto a XXX caracteres.■ Este será el comportamiento por defecto de la carga inicial de la página.

Identificador del requisito	RF02
Fuente del requisito	Aplicación web
Nombre	Buscar noticias
Descripción	La aplicación obtendrá un listado de noticias según los parámetros introducidos por el usuario en el campo de texto.
Características	<ul style="list-style-type: none"> ■ El término de búsqueda se comparará con el titular, contenido y categorías de las noticias. ■ Las noticias se agruparán en función de la relación entre ellas. ■ La búsqueda podrá ser realizada con o sin acentos y obtendrá los mismos resultados. ■ Cada noticia de la lista contendrá, al menos, el titular, la fecha de publicación, el enlace y el periódico al que pertenece.

Identificador del requisito	RF03
Fuente del requisito	Aplicación web
Nombre	Filtrar noticias
Descripción	La aplicación obtendrá un listado de noticias según los parámetros introducidos por el usuario en los distintos filtros.
Características	<ul style="list-style-type: none"> ■ Habrán filtros para el rango de fechas, las fuentes y las categorías de las noticias. ■ Las noticias se agruparán en función de la relación entre ellas. ■ Cada noticia de la lista contendrá, al menos, el titular, la fecha de publicación, el enlace y el periódico al que pertenece.

Identificador del requisito	RF04
Fuente del requisito	Aplicación web
Nombre	Cargar más caracteres
Descripción	La aplicación mostrará un botón en cada noticia para cargar más caracteres.
Características	<ul style="list-style-type: none"> ■ La aplicación cargará XXX caracteres en la noticia seleccionada. ■ El resto de noticias se verán desplazadas por el espacio que ocupe la noticia ampliada.

Identificador del requisito	RF05
Fuente del requisito	Aplicación web
Nombre	Lectura sin distracciones
Descripción	La aplicación mostrará un botón en cada noticia para ampliarla en un modal.
Características	<ul style="list-style-type: none"> ■ La aplicación cargará la noticia al completo en un modal con desplazamiento vertical propio. ■ El área destinada a la noticia será considerablemente más grande, ocupando la mayoría de la pantalla. ■ La noticia ampliada mostrará todos los elementos que se tengan sobre ella.

Identificador del requisito	RF06
Fuente del requisito	Aplicación web
Nombre	Comparar noticias
Descripción	La aplicación permitirá seleccionar noticias para su comparación.
Características	<ul style="list-style-type: none"> ■ Se podrán seleccionar de dos a cuatro noticias. ■ El botón de comparación aparecerá al seleccionar como mínimo dos noticias. ■ Las noticias seleccionadas se cargarán en un modal, compartiendo el espacio entre ellas.

Identificador del requisito	RF07
Fuente del requisito	Aplicación web
Nombre	Cambiar número de columnas de noticias
Descripción	La aplicación permitirá aumentar y disminuir el número de columnas en las que se muestran las noticias.
Características	<ul style="list-style-type: none"> ■ Se podrán seleccionar de una a cuatro columnas. ■ El ancho de cada noticia se redimensionará ocupando siempre el máximo espacio posible.

Identificador del requisito	RF08
Fuente del requisito	Aplicación web
Nombre	Indexación de noticias
Descripción	La aplicación indexará las noticias que obtenga para un rápido acceso
Características	<ul style="list-style-type: none"> ■ .

Identificador del requisito	RF09
Fuente del requisito	Aplicación web
Nombre	Mantenimiento de formato
Descripción	La aplicación mantendrá el formato original de las noticias en la visualización
Características	<ul style="list-style-type: none"> ■ Las noticias contendrán las etiquetas HTML del medio original que les dan formato.

Identificador del requisito	RF10
Fuente del requisito	Aplicación web
Nombre	Inserción de noticias
Descripción	La aplicación obtendrá noticias diariamente de distintos medios y las insertará en la BBDD
Características	<ul style="list-style-type: none"> ■ Las fuentes de las noticias quedarán definidas en un fichero externo. ■ Solo se insertarán el contenido que tenga titular, enlace a la noticia original, fecha y fuente de procedencia.

Identificador del requisito	RF11
Fuente del requisito	Aplicación web
Nombre	Facetado
Descripción	La aplicación realizará distintos facetados que permitan explorar el contenido de diversas formas
Características	<ul style="list-style-type: none"> ■ El facetado se realizará por fuentes, categorías de las noticias y fechas. ■ El resultado del facetado será utilizado para realizar una búsqueda con el término del facetado al seleccionarlo.