**Aplicación para enfermos de Alzheimer y sus cuidadores en React haciendo uso de servicios Firebase.**

Trabajo Fin de Máster

Master Universitario en Tecnologías Web, Computación en la Nube y Aplicaciones Móviles.

**Autor**: Ana Isabel Bueno Gimeno

**Tutor**: [Nombre del tutor/es]

2019-2020

Resumen

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed nisi turpis, iaculis a pulvinar quis, luctus et lorem. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Nullam vitae purus eros, id auctor dolor. Sed et nisl quis nibh fermentum cursus ut at elit. Etiam condimentum porta leo quis tempor. Quisque commodo lobortis aliquet. Etiam tincidunt, libero ut vehicula euismod, justo augue lobortis sem, et facilisis velit lacus tristique dolor.

**Palabras clave:** React, React-native, Firebase, Alzheimer, Scrum, agile.

Abstract

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed nisi turpis, iaculis a pulvinar quis, luctus et lorem. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Nullam vitae purus eros, id auctor dolor. Sed et nisl quis nibh fermentum cursus ut at elit. Etiam condimentum porta leo quis tempor. Quisque commodo lobortis aliquet. Etiam tincidunt, libero ut vehicula euismod, justo augue lobortis sem, et facilisis velit lacus tristique dolor.

**Keywords:** React, React-native, Firebase, Alzheimer, Scrum, agile.

Tabla de contenidos

[1. Introducción 7](#_Toc48236745)

[1.1. Motivación. 9](#_Toc48236746)

[1.2. Objetivos. 10](#_Toc48236747)

[1.3. Medios empleados 10](#_Toc48236748)

[1.4. Relación con las asignaturas desarrolladas en el Máster. 11](#_Toc48236749)

[2. Estado del arte 12](#_Toc48236750)

[2.1. Aplicaciones móviles en el mercado. 12](#_Toc48236753)

[2.1.1. Conclusiones. 15](#_Toc48236754)

[2.2. Tipologías de aplicaciones móviles. 16](#_Toc48236755)

[2.2.1. Aplicación nativa. 16](#_Toc48236756)

[2.2.2. Aplicaciones web o web apps 16](#_Toc48236757)

[Aplicación Web Progresiva (PWA) 17](#_Toc48236758)

[2.2.3. Aplicación híbrida 18](#_Toc48236759)

[2.2.4. Comparativa de tipologías de app. 19](#_Toc48236760)

[2.3. Lenguajes JavaScript. 19](#_Toc48236761)

[2.3.1. Marco de referencia. 21](#_Toc48236762)

[Conceptos básicos de Javascript. 21](#_Toc48236763)

[Gramatica y tipos. 21](#_Toc48236764)

[Javascript viene influenciado por la sintaxis de Java, es case-sensitive y hace uso de los caracteres Unicode. Las instrucciones se llaman sentecias y vienen separadas por ‘;’ a no ser que se escriban en una sóla línea que entonces no es necesario. Para poder hacer comentarios debe hacerse uso de // si van en una sóla línea y /\* comentario \*/ si van en varias línas, también permite el anidamiento de comentarios. Los comentarios en Javascript tienen la misma sintaxis que en C. 21](#_Toc48236765)

[Control de flujo y manejo de errores. 22](#_Toc48236766)

[Un objeto Promise puede estar en alguno de estos estados: 22](#_Toc48236767)

[Bucles e iteracciones. Javascript dispone de las siguientes sentencias para operar: 22](#_Toc48236768)

[Funciones. Las funciones son uno de los pilares fundamentales en JavaScript. Una función hace referencia a un conjunto de sentencias que realizan cualquier tipo de tarea. Para usar una función, debe definirla en algún lugar del ámbito desde el cual desea llamarla. La declaración de la misma se compone de 23](#_Toc48236769)

[Expresiones y operadores. 24](#_Toc48236770)

[Javascrip dispone de los siguientes operadores: 24](#_Toc48236771)

[Numbers and dates 24](#_Toc48236772)

[Text formatting 24](#_Toc48236773)

[Regular expressions 24](#_Toc48236774)

[Indexed collections 24](#_Toc48236775)

[Keyed collections 24](#_Toc48236776)

[Working with objects 24](#_Toc48236777)

[Details of the object model 24](#_Toc48236778)

[Using promises 24](#_Toc48236779)

[Iterators and generators 24](#_Toc48236780)

[Meta programming 24](#_Toc48236781)

[JavaScript modules 24](#_Toc48236782)

[Introducing JavaScript objects 24](#_Toc48236783)

[Client-side web APIs 24](#_Toc48236784)

[A re-introduction to JavaScript 24](#_Toc48236785)

[JavaScript data structures 24](#_Toc48236786)

[Equality comparisons and sameness 24](#_Toc48236787)

[Closures 24](#_Toc48236788)

[Inheritance and the prototype chain 25](#_Toc48236789)

[Strict mode 25](#_Toc48236790)

[JavaScript typed arrays 25](#_Toc48236791)

[Memory Management 25](#_Toc48236792)

[Concurrency model and Event LoopMotores de Javascript. 25](#_Toc48236793)

[2.3.2. Revisión de las soluciones o frameworks. 25](#_Toc48236794)

[2.3.3. Comparativa de las soluciones seleccionadas. 29](#_Toc48236795)

[2.3.4. Acerca de las solucione seleccionada. 31](#_Toc48236796)

[2.4. Bootstrap. 36](#_Toc48236797)

[Marco de referencia 37](#_Toc48236798)

[Soluciones seleccionadas. 44](#_Toc48236799)

[Comparativa de las soluciones seleccionadas. 45](#_Toc48236800)

[Acerca de las soluciones seleccionadas. 48](#_Toc48236801)

[2.5. Firebase. 50](#_Toc48236802)

[2.6. Conclusiones. 50](#_Toc48236803)

[3. Gestión del proyecto. 50](#_Toc48236804)

[3.1. Requisitos. 52](#_Toc48236805)

[3.1.1. Acceso. 53](#_Toc48236806)

[3.1.2. Usuarios. 53](#_Toc48236807)

[3.1.3. Búsquedas. 53](#_Toc48236808)

[3.1.4. Resultado de las búsquedas. 53](#_Toc48236809)

[3.1.5. Previsualizar documento. 54](#_Toc48236810)

[3.1.6. Descargar documentos. 54](#_Toc48236811)

[3.1.7. Publicación de documentos. 54](#_Toc48236812)

[3.2. Análisis. 54](#_Toc48236813)

[3.2.1. Análisis de las alternativas y selección. 54](#_Toc48236814)

[3.2.2. Casos de uso. 54](#_Toc48236815)

[3.1.1.1. Acceder al sistema. 56](#_Toc48236816)

[3.1.1.2. Buscar documento. 57](#_Toc48236817)

[3.1.1.3. Mostrar resultados. 57](#_Toc48236818)

[3.1.1.4. Abrir documento. 57](#_Toc48236819)

[3.1.1.5. Iniciar sesión. 57](#_Toc48236820)

[3.1.1.6. Publicar documentos. 57](#_Toc48236821)

[3.1.1.7. Indexar documentos. 57](#_Toc48236822)

[3.1.2. Layouts. 57](#_Toc48236823)

[3.1.3. Estimación temporal. 58](#_Toc48236824)

[3.1.4. Costes. 58](#_Toc48236825)

[3.1.4.1. Costes de material y software. 58](#_Toc48236826)

[3.1.4.2. Costes de recursos humanos. 59](#_Toc48236827)

[3.1.5. Riesgos. 59](#_Toc48236828)

[3.1.5.1. Riesgos del proyecto. 60](#_Toc48236829)

[3.1.5.2. Riesgos técnicos. 62](#_Toc48236830)

[3.1.5.3. Riesgos externos. 64](#_Toc48236831)

[3.1.5.4. Riesgos organizacionales. 65](#_Toc48236832)

[Lucene Core. 66](#_Toc48236834)

[Fragmentos y réplicaseditar 77](#_Toc48236835)

[4. Diseño. 79](#_Toc48236836)

[3.2. Análisis del Sistema. 79](#_Toc48236837)

[5. Implementación. 79](#_Toc48236838)

[6. Resultados y discusión. 80](#_Toc48236839)

[7. Conclusiones. 80](#_Toc48236840)

[8. Bibliografía 80](#_Toc48236841)

# Introducción

Actualmente, la tecnología de la información y las comunicaciones (TIC) está presente en cada uno de los sectores económicos y profesionales y se evidencia en prácticamente todos los ámbitos de nuestra vida diaria modificando nuestro comportamiento tanto laboral como social.

Por otra parte, los avances que se experimentan en todas las facetas que nos rodean hacen que el aumento de la esperanza de vida de la población se vea incrementado, esto es un reto cumplido por la sociedad, pero, ¿qué ocurre a medida que se añaden años a la vida? Existen ciertas patologías o enfermedades que se dan de forma más acentuada según los individuos alcanzan ciertas edades y existe una enfermedad que todavía, hoy en día, es imparable y afecta primordialmente a ese colectivo de personas más envejecido de la sociedad. Esta enfermedad es la demencia, trastornos neurológicos del sistema nervioso central y periférico.

Al aumentar la esperanza de vida aumenta la tasa de estos tipos de trastornos, siendo la enfermedad de Alzheimer la forma más común de demencia que según datos de la Organización Mundial de la Salud (OMS) se calcula que representa entre un 60% - 70% de los casos de demencia. Así mismo, la OMS también indica que la demencia afecta a nivel mundial a unos 50 millones de personas y prevé que en el año 2030 se llegue a los 82 millones y llegue a alcanzar los 152 millones para el 2050.

Las aplicaciones móviles pueden proporcionar a estos enfermos y sus cuidadores recursos tecnológicos para mejorar su calidad de vida, conseguir mayor autonomía para los enfermos, así como proporcionar y aportar seguridad tanto a ellos como a las personas que se encargan y preocupan por su salud. En definitiva, cubrir uno de los objetivos que tiene la OMS, añadir vida a los años.

El avance de la tecnología, redes 5G, servicios de posicionamiento y localización, etc. Hacen que el desarrollo de aplicaciones para estos colectivos sea además de necesario y útil para su día a día, posible.

Pero ¿Qué es la enfermedad del Alzheimer? El Alzheimer es un tipo de demencia. La palabra demencia proviene del latin dementia que significa sin mente, la OMS la describe como “*La demencia es un síndrome –generalmente de naturaleza crónica o progresiva– caracterizado por el deterioro de la función cognitiva (es decir, la capacidad para procesar el pensamiento) más allá de lo que podría considerarse una consecuencia del envejecimiento normal. La demencia afecta a la memoria, el pensamiento, la orientación, la comprensión, el cálculo, la capacidad de aprendizaje, el lenguaje y el juicio. La conciencia no se ve afectada. El deterioro de la función cognitiva suele ir acompañado, y en ocasiones es precedido, por el deterioro del control emocional, el comportamiento social o la motivación.*” El Alzheimer representa el tipo más común de demencia, aunque pueden existir otros tipos de demencias como los derivados de ictus, o problemas vasculares comúnmente se engloba a todos ellos como Alzheimer.

La enfermedad del Alzheimer toma el nombre por el psiquiatra y neurólogo alemán Alois Alzheimer en 1906 cuando identifica que una paciente presenta problemas de sueño, de memoria, cambios de humor y una confusión gradual. Al fallecer, el doctor le realizó la autopsia y en su análisis comprobó que existían irregularidades en el tejido cerebral. Se trata de una demencia neurodegenerativa donde las neuronas se van desgastando y muriendo de forma lenta y progresiva. Esta muerte neuronal va atrofiando diferentes partes del cerebro y hoy en día es una enfermedad incurable y finalmente mortal. Habitualmente la enfermedad se presenta en etapas avanzadas de la vida, pero existen casos donde aparece personas relativamente jóvenes.

La enfermedad comienza con una reducción paulatina de la capacidad intelectual, un deterioro cognitivo que poco a poco va afectando a la capacidad del desarrollo de las actividades de la vida diaria. El Alzheimer afecta a:

* Atención
* Percepción
* Memoria
* Pensamiento
* Lenguaje
* Emotividad
* Comportamiento
* Inteligencia

En un primer momento el enfermo va olvidando las cosas cotidianas de la vida como el nombre de objetos o personas, perderse en lugares o sitios familiares, cambios de humor o de personalidad, todo esto hace que los que le rodean perciban una alteración en su comportamiento. Al principio esto ocurre de forma puntual sin ser conscientes ni enfermo ni familiares de estos cambios y finalmente es familia o personas muy allegadas quienes empiezan a notar estos síntomas.

Un enfermo de Alzheimer pasa por distintas fases pudiéndose distinguir las siguientes tres:

Fase 1

De 1 a 3 años alteraciones de memoria y de la orientación espacial. En este momento es difícil ser consciente de percibir estos cambios. Olvido de tareas, repetir las mismas preguntas, cierta desorientación, prevalencia de los recuerdos antiguos sobre los recientes, cambios en la personalidad como tristeza, irritación, ansiedad o insomnio son los síntomas que empiezan a aparecer

Aunque se conserva la capacidad de mantener una conversación se aprecia un déficit de palabras, dificultad para orientarse en espacios poco conocidos y progresivamente en espacios familiares.

Fase 2

Entre 2 y 10 años. Se aprecian los síntomas anteriores de forma mucho más acusada, ya no son tan leves lo que hace evidente que el problema está presente. Uno de los elementos típicos de esta fase es que comienzan a aparecer alteraciones más profundas del lenguaje y del sistema motor como pueden ser dificultades al caminar, rigidez o debilidad muscular y la tendencia del enfermo es permanecer la mayor parte tiempo tumbado o sentado.

Fase 3

Se da entre los 8 y 12 años con un deterioro mental grave. La persona se encuentra totalmente apática y se pierde incluso la capacidad de realizar conductas automáticas (comer, beber, andar, aparece incontinencia) hasta que finalmente el enfermo acaba postrado en la cama.

Con todo esto, el proyecto que se presenta en este documento Trabajo Fin de Master (TFM) representa el entregable de la asignatura ‘2019-20 Treball Fi de Màster Gr.PT (44836)’ de carácter obligatorio para la finalización del máster ‘Máster en Tecnologías Web, Computación en la Nube y Aplicaciones Móviles’ y pretende realizar una aplicación que pueda ser usada por enfermos y cuidadores para ayudar a todos los implicados en el seguimiento y control de la enfermedad principalmente en las fases 1 y 2 de la misma aportando mayor calidad de vida.

INDICAR LA ESTRUCTURA DE LA MEMORIA DEL TFM.

## Motivación.

La sociedad está cada vez más envejecida y con más frecuencia se dan distintos tipos de demencia en la población y entre uno de ellos se encuentra el Alzheimer que representa el tipo con mayor tasa. El Alzheimer es una enfermedad que roba la memoria, es un ladrón los de los recuerdos vividos por la persona que lo padece, llega sin avisar, destroza una vida y en muchos casos llega a destrozar una familia.

Los cuidadores necesitan mucho apoyo y comprensión por eso he querido poner mi pequeño grano de arena y sumar mediante la creación de esta aplicación. La aplicación que de este proyecto surja considero que puede ser de utilidad tanto al propio enfermo en sus fases iniciales de la enfermedad, así como a los cuidadores.

Por fortuna no he vivido de forma personal ningún familiar que la haya padecido, pero sí he sido voluntaria de distintas asociaciones con lo que he vivido la enfermedad de forma muy próxima sintiendo mucha sensibilidad hacia ella y quien la padece de uno u otro modo.

El objetivo es unir las nuevas tecnologías a los servicios y cuidados que necesita la persona enferma. Aunque en la actualidad existen aplicaciones móviles que abarcan parte de la funcionalidad que se pretende no se ha encontrado una que cubra todas ellas. Las aplicaciones del mercado contemplan las siguientes funcionalidades:

* Herramientas de información y apoyo, resolviendo dudas y preguntas de cuidadores y familiares.
* Juegos, entretenimiento y ejercicios para el refuerzo de memoria.
* Recordatorios.
* Conocer la ubicación de la persona enferma.
* Llamadas.

La aplicación que se pretende desarrollar unirá todas estas funcionalidades en una única aplicación y añadirá otras nuevas como el aviso a los cuidadores y/o familiares cuando el enfermo salga de la zona de seguridad establecida por ellos mismos.

## Objetivos.

El objetivo principal del proyecto es desarrollar una aplicación multiplataforma, esto es, que pueda ser ejecutada desde la web así como para dispositivos móviles de sistemas Android e iOs mediante la tecnología React y haciendo uso de los servicios en la nube que ofrece Firebase.

Esta aplicación será de ayuda tanto para enfermos de Alzheimer (extensible a otros tipos de demencia) como a sus cuidadores y/o familiares.

Otros objetivos más específicos del proyecto contemplan:

* Desarrollo de la aplicación multiplataforma.
  + iOs
  + Android
  + Web
* Visión distinta de la aplicación en función del rol de acceso.
  + Pantallas acondicionadas para enfermos de Alzheimer.
  + Pantallas acondicionadas para cuidadores y/o familiares.
* Uso de los servicios ofrecidos por Firebase:
  + Base de datos para realizar la persistencia de estos.
  + Geolocalización.
  + Login/autenticación seguro.
* Accesibilidad de la aplicación para los enfermos con disfunciones funcionales.
* Gestión del proyecto de forma ágil utilizando metodología Scrum en lugar del clásico proyecto en cascada.

## Medios empleados

REVISAR EN LA FINALIZACION DEL PROYECTO

Para la realización de la aplicación se han utilizado las siguientes herramientas:

* Git: Herramienta de software libre para el control de versiones del código fuente de la aplicación distribuido. También se ha utilizado para el versionado del documento de la memoria del TFM.
* Git-hub: Sistema de control de versiones que se apoya en Git, gestión de proyectos y repositorio de trabajo colaborativo. El código fuente y documentación generada en el proyecto se almacena en este repositorio online.
* Visual Studio Code: Herramienta de edición de código fuente al que pueden integrarse numerosos plugins o complementos con el fin de ayudar y facilitar el desarrollo como Git o depuración de código.

Y se ha hecho uso de las siguientes librerias y APIs

* React: Librería Javascript para el desarrollo de aplicaciones web y móvil gracias al XXXX. Existe un ecosistema entorno a react con módulos, herramientas y componentes ‘pre construidos’ o listos para facilitar al desarrollador las funcionales a implementar.

Por último, para la gestión del proyecto y documentación:

* Microsoft Word: Para la redacción de la memoria.
* Git-hub: Para el uso de la wiki y tablero SCRUM.
* Mockflow: Utilizado para la generación de wilframes o diseños de las pantallas.
* Hungouts para la comunicación con el tutor externo.

## Relación con las asignaturas desarrolladas en el Máster.

PUNTO AL FINALIZAR LA MEMORIA.

01.- Administración de recursos y automatización de operaciones.

02.- Centros de Datos y virtualización.

03.- Programación del lado cliente.

04.- Programación del lado servidor.

05.- Gestión y distribución de contenido multimedia.

06.- Métodos de producción de software.

07.- Seguridad.

08. Análisis de datos Web i sociales.

09. Computación en la nube.

10. Dispositivos móviles y realidad aumentada.

11. Persistencia relacional i no relacional datos.

12. Desarrollo Basado en Componentes Distribuidos y Servicios.

# Estado del arte

Con el fin de abordar el presente trabajo ha sido necesario llevar a cabo un proceso de análisis de la situación actual de las aplicaciones existentes en el mercado como de las tecnologías que van a ser utilizadas, lo más innovador, reciente o con mayor tendencia actual. Además, se pretende recoger una definición e introducción a algunos conceptos a modo de contextualizar el Trabajo Fin de Máster.

En este apartado del estado del arte, se recogen las referencias más significativas empleadas para profundizar en el conocimiento de las soluciones.

Para todos los elementos que componen o forman parte de este trabajo existe una gran cantidad de soluciones y alternativas disponibles; lo que evidencia la necesidad de acotar el estudio del problema, apoyándose en los resultados obtenidos en las obras referenciales más significativas.

Teniendo como objetivo del proyecto la creación o construcción de una aplicación que se comporte de forma adecuada tanto en dispositivos móviles (smartphones y tablets) como en ordenadores; se ha seleccionado la tecnología más idónea para que pudiera soportar este requisito. Frente a tener una aplicación móvil + desarrollar una aplicación web se ha optado por la construcción de una aplicación web progresiva (PWA) que permite la coexistencia para cualquier dispositivo que tenga instalado un navegador web.



## Aplicaciones móviles en el mercado.

A continuación, se detallan las aplicaciones móviles para Android o IOs encontradas en los Stores de Google y Apple. Aunque existen muchas aplicaciones, podemos agruparlas según su funcionalidad o el objetivo que buscan.

Aplicaciones divulgativas y asesoramiento. El objetivo de estas aplicaciones son divulgar, aconsejar, informar y dar a conocer la enfermedad, están orientadas a cuidadores y/o familiares de pacientes con este diagnóstico. Podemos encontrar, Pharma Alzheimer, Alzheimer's Daily Companion, YoTeCuido Alzheimer, CRE de Alzheimer Imserso, Alzheimer's Assoc Meetings, etc.

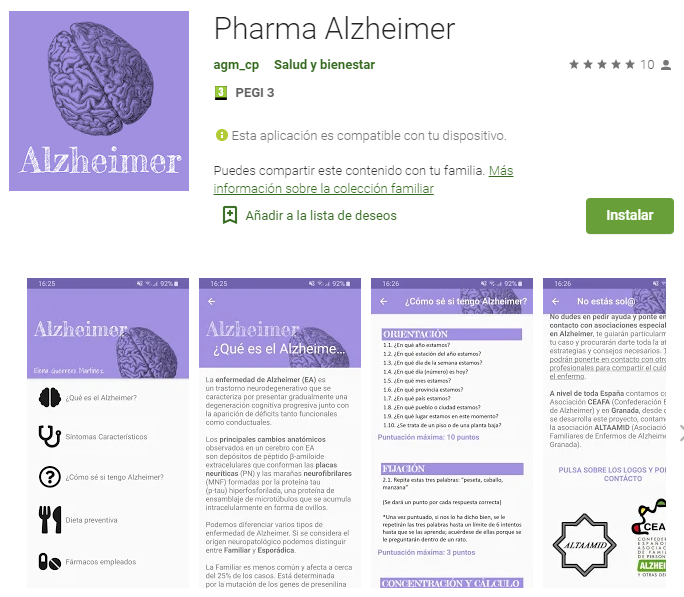
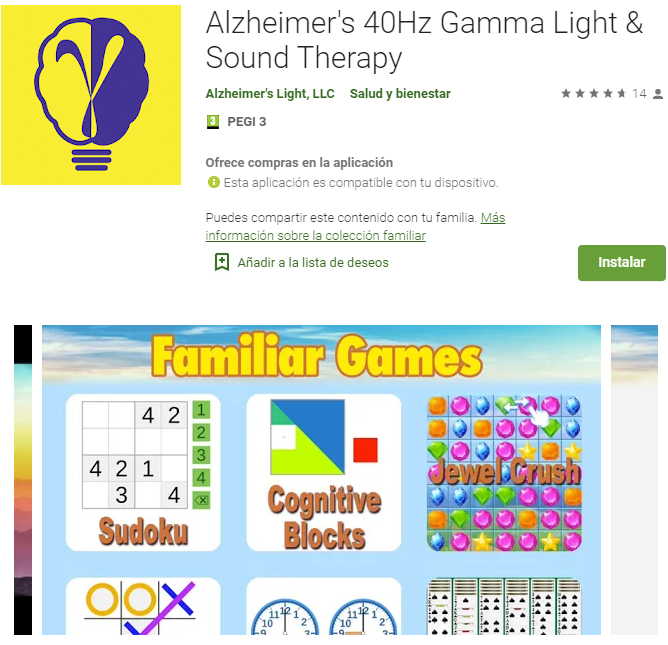
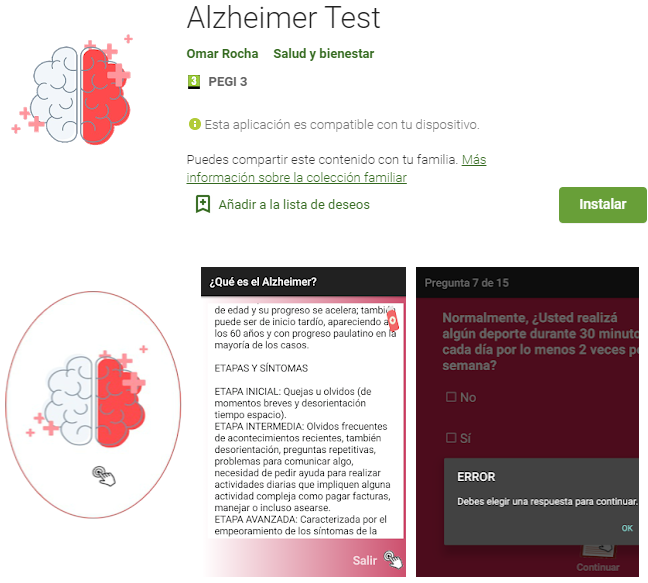


Ilustración 1 - https://play.google.com/store/apps/details?id=com.egm.PharmAlzheimer

Juegos y ejercicios: Aplicaciones concebidas para ejercitar la memoria, las asociaciones de objetos u elementos. Adivinanzas, ejercicios matemáticos, asociaciones de pares de objetos Este conjunto de aplicaciones es el más amplio en las plataformas de markets para descargas, así encontramos aplicaciones como pueden ser Alzheimer Info y Ejercicios, NoMeOlvides, Alzheimer's 40Hz Gamma Light & Sound Therapy, Ejercicio de memoria para el Alzheimer, Entrena tu Cerebro - Juegos de Memoria, Alzheimer's Speed of Processing Game - ASPEN, Juegos de Alzheimer, Match, etc.



Tests o evaluaciones de la enfermedad: estas aplicaciones tienen como objetivo realizar un primer análisis sobre el estado de la enfermedad en la persona que lo está realizando. Encontramos aplicaciones como Dementia & Alzheimer's Memory Diagnosis Test: MMSE o Alzheimer Test



## Conclusiones.

Al realizar una búsqueda en el Market place tanto de Google como de iPhone se pueden encontrar aplicaciones que intentan dar solución a la temática del Azheimer pero después de hacer una revisión de todas las que están disponibles, se observa que prácticamente toda la totalidad de las aplicaciones relacionadas con esta enfermedad van dirigidas a juegos o ejercicios, algunas a divulgar e informar sobre la misma y ninguna dedicada a poder hacer seguimiento de los pacientes por parte de los cuidadores sean estos profesionales o cuidadores.

Este tipo de aplicaciones son poco populares y con bajas descargas con lo que la inversión en ellas es muy poca, muchas de ellas dejan de funcionar al no ser mantenidas y la evolución tecnológica que se va produciendo en las distintas versiones de los dispositivos.

Es por todo ello que se hace más motivador poder crear una aplicación que aúne todas estas funcionalidades, información, ejercicios y seguimiento de los pacientes.

## Tipologías de aplicaciones móviles.

A continuación se van a detallar las tres tipologías de aplicaciones móviles en las que puede englobarse cualquier aplicación:

## Aplicación nativa.

Aunque es complicado determinar exactamente el primer móvil que disponía de una aplicación móvil o “app” sí puede decirse que, de forma general, las primeras aplicaciones de las que disponían los teléfonos móviles estaban relacionadas con la gestión y organización del trabajo como son agenda, bloc de notas, calculadora, base de datos de contactos. Se trataban de aplicaciones con una funcionalidad simple y elemental. Con la revolución de los teléfonos inteligentes en el 2007 y la aparición de las tiendas virtuales en 2008 para las descargas de aplicaciones como Play Store de Google para sistema operativo Android o App Store de Apple para dispositivos iOs, la creación de este tipo de aplicaciones se incrementa.

Las aplicaciones nativas, no son más que aplicaciones add-hoc, es decir, debe realizarse de forma específica una aplicación para el sistema operativo del dispositivo donde va a ser instalada para que esta funcione. Así pues, la construcción de una aplicación que quiera ejecutarse en Android, de forma general, seria desarrollada en Java o Kotlin (aunque existen otros lenguajes de programación), se programaría en Swift u Objective-C si quiere desplegarse en iOs y por ejemplo en C# o .Net si lo que querríamos es una aplicación para Windows. El código fuente de estas aplicaciones se compila a un ejecutable donde también se incluyen los recursos multimedia utilizados (imágenes, vídeos, sonidos…) y se deja disponible en los markets para que sean descargados por los usuarios.

## Aplicaciones web o web apps

La aplicación web o web apps es aquella aplicación que hacen uso del conjunto de las típicas tecnologías web siendo estas Javascript, HTMLy CSS para su desarrollo. Al contrario de las aplicaciones nativas, con la construcción de una sola aplicación en estos lenguajes de programación o estilos se consigue que puedan ser ejecutadas sólo teniendo instalado un navegador web en el dispositivo. Estas aplicaciones pueden ser ejecutadas en cualquier tipo de dispositivo adaptándose a las distintas pantallas, dimensiones y tamaños de cada terminal.

Desde la década de los 90, donde se comenzó con la generación de **páginas web**, hasta el momento actual con las **web apps**, ha habido una evolución importante en todo lo relativo al mundo web. En sus momentos iniciales, el contenido de los sitios era básicamente texto y colores; su uso se limitaba a realizar una comunicación funcional e informativa más que de entretenimiento.

Con la introducción del lenguaje de hipertexto más avanzado como las tablas y el surgimiento del javascript en 1995 dando resolución a las limitaciones que tenía el HTML se observa una evolución importante tanto en el diseño como en el contenido de los sitios web. A finales de los 90 aparece el lenguaje de estilos CSS, las hojas de estilo en cascada, dividiendo o separando la página en la parte de contenido que es gestionada por el HTML y la parte de forma y apariencia que es trabajada por CSS.

Los diferentes navegadores y sus versiones van dificultando el trabajo del desarrollador puesto que la visualización y comportamiento de las páginas era diferente en función de lo que pudieran soportar.

A principios de 2003 y con la salida de la web 2.0 se pone atención en el usuario, dando importancia a su experiencia de navegación y usabilidad. Las interfaces se focalizan en ser más atractivas y el diseño cobra importancia.

El siguiente paso importante que supuso una evolución en esta tecnología web fue la aparición de los teléfonos móviles inteligentes “smartphones” aunque ya aparece algún modelo en 1994, no es hasta finales de los 90 y principios de los 2000 cuando se lanza el conjunto de protocolos que permiten ver en los dispositivos móviles el contenido de las páginas web adaptadas, se trata de la tecnología WAP (Wireless Application Protocol), en este momento es cuando realmente comienza su gran progreso y es Steve Jobs en 2007 cuando da a conocer su iPhone siendo éste hecho la verdadera revolución. En la presentación del iPhone reflexiona sobre qué ocurrirá con los desarrolladores explicando que tienen una nueva e innovadora forma de crear aplicaciones móviles, a través del motor de Safari “*puedes escribir la increíble* ***web 2.0 y Ajax******que se ve y se comporta exactamente como las aplicaciones del iPhone*** *y estas aplicaciones se pueden integrar perfectamente con los servicios del iPhone pueden hacer una llamada, pueden enviar un correo electrónico, pueden buscar una ubicación en Google Maps. Después de escribirlas tienes distribución instantánea no tienes que preocuparte por la distribución solo ponlas en tu servidor de Internet y son realmente fáciles de actualizar, solo cambia el código en tu propio servidor en lugar de tenerlo a través de este proceso de actualización realmente complejo y están asegurados con el mismo tipo de seguridad que se utiliza para las transacciones con Amazon o un banco y se ejecutan de forma segura en el iPhone para que no comprometan su fiabilidad o seguridad y piensa que no hay necesidad de un SDK para aplicaciones nativas. Tiene todo lo que necesita si usted sabe cómo escribir aplicaciones utilizando los estándares web más modernos para escribir aplicaciones increíbles*”. Aquí, Steve Jobs presentó la idea visionaria de las aplicaciones web.

En 2010 se produce el último gran cambio en el diseño web llegando el diseño responsivo o adaptable dando solución a que un sitio web se adaptase a las dimensiones de los distintos dispositivos inteligentes, logrando hacer que todos estos sitios se presentasen al usuario de una forma más amigable generando una mejor experiencia de usuario y navegación.

Finalmente, esta idea visionaria de aplicaciones web no evoluciona en demasía hasta 2015, donde Alex Russel desarrollador de Chrome acuña el término “**Web progressive applications** (**WPA**)” en un artículo “Progressive Web Apps: Escaping Tabs Without Losing Our Soul”

Este tipo de aplicaciones están cada vez más en auge, son aplicaciones web que se adaptan a cada dispositivo de forma progresiva basadas en tecnologías web y sus lenguajes HTML5, CSS y JavaScript junto con un navegador consiguen acceder a las funcionalidades y sensores disponibles en los dispositivos móviles para extraer todo su potencial e incrementar la mejora en la experiencia del usuario.

Este tipo de aplicaciones son capaces de utilizar las funcionalidades y características que actualmente poseen los navegadores de última generación proporcionando una experiencia al usuario muy similar a las aplicaciones nativas de los teléfonos móviles con la ventaja de que, con el desarrollo de una sola aplicación, ésta, puede ejecutarse en los dispositivos más utilizados actualmente, móviles, tablets, ordenadores portátiles, fijos o de sobremesa. De este modo ya no es necesario desarrollar una aplicación móvil para Android/iOs y otra para el acceso desde ordenador.

En conclusión, una web app progresiva sería una evolución entre las páginas web y una aplicación nativa, aportando al usuario final las ventajas de éstas, pero haciendo uso de los recursos que ofrecen los navegadores de última generación y las capacidades del dispositivo móvil adaptándose a él (de forma progresiva).

## Aplicación híbrida

Las aplicaciones híbridas son una combinación de aplicaciones nativas y las aplicaciones web. Utilizan las virtudes de las aplicaciones web responsivas, es decir, una única aplicación podría ejecutarse en cualquier teléfono móvil con independencia de su sistema operativo y las virtudes de las aplicaciones nativas, que sin hacer desarrollos específicos para cada sistema operativo se adaptan a él para acceder y aprovechar todo el potencia del dispositivo móvil como son las notificaciones push, la cámara, GPS, sensores del dispositivo, etc aunque su rendimiento es algo menor a las nativas.

Este tipo de aplicaciones utilizan una capa de abstracción sobre la plataforma nativa. La base tecnológica de estas aplicaciones son las tecnologías web, los lenguajes web HTML5, JavaScript y CSS junto con un framework que es esa capa de abstracción que permite adaptar la vista web a la vista de los terminales móviles. Dentro de estos marcos de trabajo multiplataforma podemos encontrar Ionic, React Native, Xamarint, Flutter, PhoneGap o JQuery Movile, etc.

Estos frameworks hacen uso de APIs (Interfaz de Programación de Aplicaciones) que permiten acceder y hacer uso de los elementos hardware del dispositivo (cámara, acelerómetro, gps, etc.) y al igual que las aplicaciones nativas, el código fuente de la aplicación se empaqueta con la ayuda del webkit proporcionado por estos frameworks para su posterior publicación en los markets de cada plataforma Play Store (Google) o App de iOs (Apple)..

## Comparativa de tipologías de app.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Nativa | Aplicación web | Híbrida |
| Experiencia de usuario | Alta | Medio - Alto | Medio-Alto |
| Rapidez de desarrollo | Alto (aunque una aplicación por SO) | Alto |  |
| Coste | Alto | Medio | Alto |
| Usabilidad | Alta | Alta | Media-Alta |
| Conexión a internet | No necesario | Necesario | No necesario |
| Rendimiento | Alto | Alto | Medio-Alto |
| Independencia de SO | Bajo | Alto | Medio |
| Despliegue y publicación | En app store | Rápido. No requiere aprobación externa | En app store |

## Lenguajes JavaScript.

En sus inicios la web únicamente era lenguaje de marcas de hipertexto (HTML) y el contenido de esta era básicamente textos, imágenes, enlaces. No había ningún tipo de dinamismo en las páginas que se creaban. También esto era así debido a la lentitud de la conexión y comunicaciones del momento; en la década de los 90, los pocos usuarios que se conectaban a internet lo hacían con módems a una velocidad de 28.8 kbps (kilobites por segundo). La infraestructura de la *World Wide Web* por aquel momento no permitía mucho tráfico de datos entre el navegador web y el servidor.

El avance en el desarrollo de páginas web hacia aplicaciones web que proporcionaban más funcionalidad y características a los usuarios y más complejidad en las mismas surge la necesidad de que hubiera un lenguaje de programación que se ejecutase en el lado del cliente, del navegador, para reducir ese tráfico de datos entre cliente-servidor y mejorar la experiencia del usuario, así como la sensación de lentitud al acceder a estas webs.

A parte de estas características cabe mencionar que sólo existían dos navegadores Netscape (hoy en día Mozilla Firefox) e Internet Explorer de Microsoft, cada uno de ellos quería ser el pionero en inducir la corriente principal de desarrollo sobre los navegadores.

En esta lucha entre competidores, aparece Brendan Eich, un programador de Netscape. Este pensó que podría solucionarse este problema adaptando otras tecnologías existentes (como *ScriptEase*) al navegador Netscape Navigator 2.0, que iba a lanzarse en 1995. Inicialmente, Eich denominó a su lenguaje *LiveScript*. Aprovechando el auge y popularidad que tenía el lenguaje de programación Java en este momento y la alianza surgida entre Sun Microsystems (creador de Java) y Netscape, provocan que, por motivos de marketing, se le cambiase el nombre a JavaScript. Esto también supuso que, durante mucho tiempo, muchos confundieran ambos lenguajes de programación.

Mientrastanto, Microsoft avanza en su lenguaje de programación para su navegador Internet Explorer al que le llama JScript, evitando así problemas legales, pero que no supone más que una ‘copia’ del lenguaje de JavaScript de Netscape.

En este momento, Netscape decide estandarizar el lenguaje Javascript. En 1997 envía la versión de JavaScript 1.1 a la asociación ECMA (*European Computer Manufacturers Association*) para tal fin.

ECMA creó el comité TC39 con el objetivo de *"estandarizar de un lenguaje de script multiplataforma e independiente de cualquier empresa"*. El primer estándar que creó el comité TC39 se denominó **ECMA-262**, en el que se definió por primera vez el lenguaje ECMAScript. Y más tarde, la organización internacional para la estandarización (ISO) adoptó el estándar ECMA-262 a través de su comisión IEC, dando lugar al estándar ISO/IEC-16262.

ECMA ha publicado varios estándares relacionados con ECMAScript. Desde junio de 1997 cuando se publicó la primera edición del estándar ECMA-262 hasta Junio de 2020 se han publicado 11 ediciones, la última ECMAScript ® 2020 Especificación del Lenguaje (Esta Norma define el lenguaje de programación de propósito general ECMAScript 2020). Estas versiones también reciben el nombre de ES por el acrónimo de ECMAScript y van desde la ES1 en junio de 1997 hasta la ES6, edición de 2014. A partir de este momento se nombra con el año en el que sale la edición comenzando en ES2015 hasta día de hoy ES2020

## Marco de referencia.

Según MDN Web Doc, plataforma de aprendizaje para las tecnologías Web y el software con el que funciona la Web, "JavaScript® es el lenguaje interpretado orientado a objetos desarrollado por Netscape que se utiliza en millones de páginas web y aplicaciones de servidor en todo el mundo. JavaScript de Netscape es un superconjunto del lenguaje de scripts estándar de la edición de ECMA-262 3 (ECMAScript) que presenta sólo leves diferencias respecto a la norma publicada."

JavaScript puede decirse que es un lenguaje de programación dinámico que soporta construcción de objetos [basado en prototipos](https://wiki.developer.mozilla.org/en/JavaScript/Guide/Details_of_the_Object_Model#Class-Based_vs._Prototype-Based_Languages) que a diferencia de los lenguajes de programación orientados a objetos en los que se tienen clases e instancias de estas clases, este tipo de programación simplemente tiene objetos, estos son los *objetos prototípicos que son objetos que se utilizan como plantillas a partir de la cual se obtienen el conjunto de propiedades de un nuevo objeto. Un objeto cualquiera puede especificarse sus propias propiedades (en la creación o en tiempo de ejecución) pero además puede ser utilizado como prototipo de otro objeto compartiendo sus propiedades, estas porpiedades pueden ser añadidas o modificadas en tiempo de ejecución*. Así pues, este lenguaje puede funcionar como [lenguaje procedimental](http://www.instantweb.com/~foldoc/foldoc.cgi?query=procedural&action=Search), donde el programador wspecifica la secuencia pasos a seguir  y como [lenguaje orientado a objetos](http://www.instantweb.com/~foldoc/foldoc.cgi?query=object+oriented&action=Search).

JavaScript contiene una librería estándar de objetos (Array, Date y Math) y elementos centrales del lenguaje (operadores, estructuras de control, y sentencias). El núcleo de JavaScript puede usarse en varios propósitos:

*Client-side JavaScript* extiende el núcleo del lenguaje proporcionando objetos para controlar un navegador y su modelo de objetos (o DOM, por las iniciales de Document Object Model).

*Server-side JavaScript* extiende el núcleo del lenguaje proporcionando objetos relevantes a la ejecución de JavaScript en un servidor.

### Conceptos básicos de Javascript.

### Gramatica y tipos.

### Javascript viene influenciado por la sintaxis de Java, es case-sensitive y hace uso de los caracteres Unicode. Las instrucciones se llaman sentecias y vienen separadas por ‘;’ a no ser que se escriban en una sóla línea que entonces no es necesario. Para poder hacer comentarios debe hacerse uso de // si van en una sóla línea y /\* comentario \*/ si van en varias línas, también permite el anidamiento de comentarios. Los comentarios en Javascript tienen la misma sintaxis que en C.

Existen tres tipos de declaraciones en Javascript

* var : Declara una variable, inicializándola opcionalmente a un valor.
* let: Declara una variable local en un bloque de ámbito, inicializándola opcionalmente a un valor.
* const: Declara una constante de sólo lectura en un bloque de ámbito.

Las variables pueden ser globales o locales y tienen la propiedad **hoisting** es decir, pueden hacer referencia a una variable declarada más tarde,  sin obtener una excepción aunque sí puede obtenerse un ‘undefined’.

JAvascript tiene un total de 8 tipos de datos, 7 son primitivos estos son: Boolean, null, undefined, Number, BigInt, String, Symbol y el tipo de datos Objet.

### Control de flujo y manejo de errores.

Para dar mayor interactividad a las aplicaciones, Javascript soporta un conjunto compacto de sentencias, específicamente para el manejo de flujo. A continuación, se detallan.

Sentencia de bloque. Son las más básicas y se expresan todas las sentencias necesarias en un bloque delimitada con un par de llaves. Se inicia con llave abierta { y se cierra con la llave cerrada }

Sentencias condicionales. Es un conjunto de comandos que se ejecutan si una condición es verdadera. JavaScript soporta dos sentencias condicionales: if...else y switch

Sentencias para el manejo de excepciones. Se pueden lanzar excepciones usando la sentencia throw y manejarlas o capturarlas usando las sentencias try...catch gestionando los errores que se recogen como objeto. Javascript también dispone del bloque finally que se ejecutará después del manejo de la excepción.

Promise

Las promesas aparecen a partir de ECMAScript 6 (ES6), con ellas Javascript gana soporte para los objetos Promise que le permiten controlar el flujo de operaciones diferidas y asíncronas.

### Un objeto Promise puede estar en alguno de estos estados:

* pending: Estado inicial, no terminada exitosamente o rechazada.
* fulfilled: operación exitosa.
* rejected: operación fallida.
* settled: la Promesa ha sido exitosa o fallida, pero no está pendiente.

### Bucles e iteracciones. Javascript dispone de las siguientes sentencias para operar:

* sentencia for. se repite hasta que la condición especificada se evalúa como false
* sentencia do...while. se repite hasta que la condición especificada que se evalúa sea false, en este caso, la primera sentencia no será evaluada con lo que siempre será ejecutada.
* sentencia while. Ecuta la/s sentencia/s del bloque mientras la condición sea evaluada como verdadera.
* sentencia label. proporciona una sentencia con un identificador que permite referirse a él desde cualquier lugar de su programa
* sentencia break. Se hace uso de esta sentencia para salir de un bucle, switch, o de una sentencia label.
* sentencia continue. Para reiniciar una sentencia while, do-while, for, o label.
* sentencia for...in.  Con esta sentencia se itera una variable enumerable de un objeto
* sentencia for...of. crea un bucle iterando sobre objetos (Array, Map, Set…)

### Funciones. Las funciones son uno de los pilares fundamentales en JavaScript. Una función hace referencia a un conjunto de sentencias que realizan cualquier tipo de tarea. Para usar una función, debe definirla en algún lugar del ámbito desde el cual desea llamarla. La declaración de la misma se compone de

* La palabra reservada  [function](https://wiki.developer.mozilla.org/en-US/docs/JavaScript/Reference/Statements/function" \o "function), seguida por:
* El nombre de la función (opcional).
* Una lista de argumentos para la función, encerrados entre paréntesis y separados por comas (,).
* Las sentencias JavaScript que definen la función entre llaves { }

Las funciones en Javascript permiten, anidamiento, recursividad y cierres (closure)

Las funciones contienen el objeto arguments con estructura similar al array aunque no lo es. Este objeto contiene un conjunto de los argumentos que le pueden ser pasados a la función, su beneficio es que se puede llamar una función con más argumentos de los que inicialmente fueron declarados que puede aceptar.

Parámetros de la función y parámetros rest, los parámetros de la función son los que se le pasan cuando se declara la misma, sin embargo, los parámetros rest nos ayudan a no tener que definirlos todos, es decir, supone respresentar un número indefinido de argumentos en forma de array; su sintaxis es …nombreArgumento.

Funciones flecha. Se introdujeron a partir de ES6 por varios motivos, acortar su sintaxis y no tener this, arguments, super o new.target. Estas funciones son siempre anónimas.

Otras funciones Javascript predefinidas de nivel superior.

* eval. evalúa el código JavaScript representado como un string
* isFinite. Comprueba si el valor pasado es infinito
* isNaN. Comprueba si el valor pasado es numérico.
* parseInt and parseFloat. Devuelve el valor numérico en cada caso al pasarle una cadena
* Number and String. Convierten una cadena a un número y viceversa.
* encodeURI, decodeURI, encodeURIComponent, and decodeURIComponent. A partir de ES5 aparecen estas funciones para codificar y decodificar cadenas

### Expresiones y operadores.

### Javascrip dispone de los siguientes operadores:

* asignación
* comparación
* aritméticos
* bit a bit
* lógicos
* de cadena de caracteres
* condicional (ternario)
* coma
* unarios
* relacionales

### Numbers and dates

### Text formatting

### Regular expressions

### Indexed collections

### Keyed collections

### Working with objects

### Details of the object model

### Using promises

### Iterators and generators

### Meta programming

### JavaScript modules

### Introducing JavaScript objects

### Client-side web APIs

### A re-introduction to JavaScript

### JavaScript data structures

### Equality comparisons and sameness

### Closures

### Inheritance and the prototype chain

### Strict mode

### JavaScript typed arrays

### Memory Management

### Concurrency model and Event LoopMotores de Javascript.

## Revisión de las soluciones o frameworks.

### 

Dentro del contexto de Javascript existen numerosos frameworks de trabajo algunos de ellos más en auge y populares que otros. Según la encuesta de ‘The Stata of JavaScript’ que encuesta a desarrolladores de todo el mundo. Contestaron sobre el uso que hacían de los frameworks; en primer lugar por uso estaría React, seguido de Vue y como tercer framework más usado Angular.

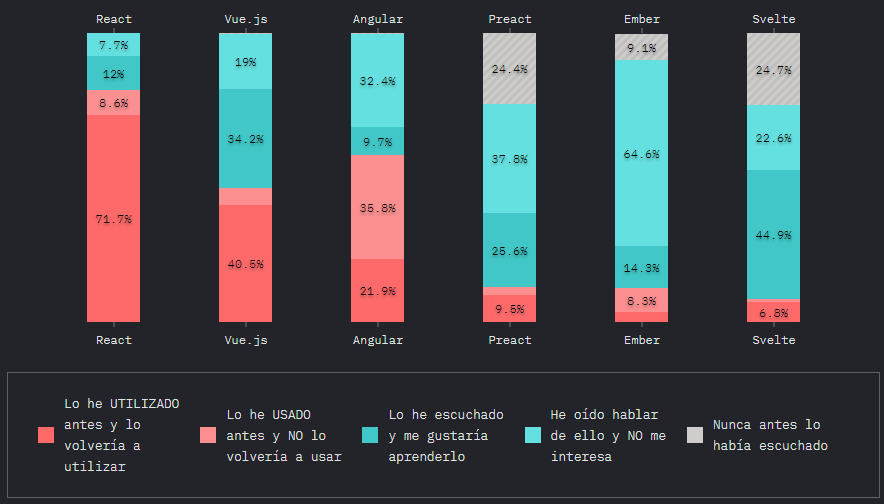


Ilustración 2: Clasificación basada en The State of JavaScript

<https://2019.stateofjs.com/front-end-frameworks/>

En base a la encuesta anual para desarrolladores de Stack Overflow en la que en 2019 63,585 desarrolladores respondieron a sus preguntas sobre el frameworks de javascript más utilizado era en este orden, jQuery, React y Angular.

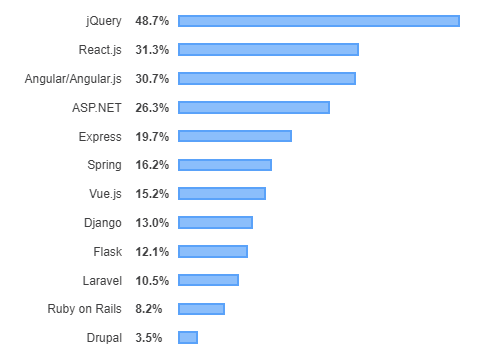


Ilustración 3: framewoks más usados según la encuesta de Stack Overflow <https://insights.stackoverflow.com/survey/2019/#technology-_-web-frameworks>

En cuanto al más amado, la respuesta de los 58,543 que participaron era por este orden, React, Vue y ángular A continuación se muestran la grafícas con los resultados a la encuesta.

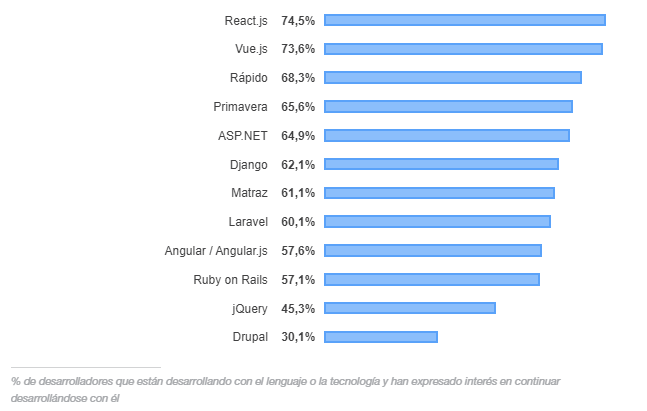


Ilustración 4: framewoks más amados según la encuesta de Stack Overflow <https://insights.stackoverflow.com/survey/2019/#technology-_-most-loved-dreaded-and-wanted-web-frameworks>

Por último, si observamos las tendencias que arroja NPM donde se contabilizan los recuentos de descargas de paquetes a lo largo del tiempo se observa, como puede verse en la siguiente gráfica, que la tendencia durante estos últimos 5 años es react quién saca ventaja sobre vue, angular o jQuery.

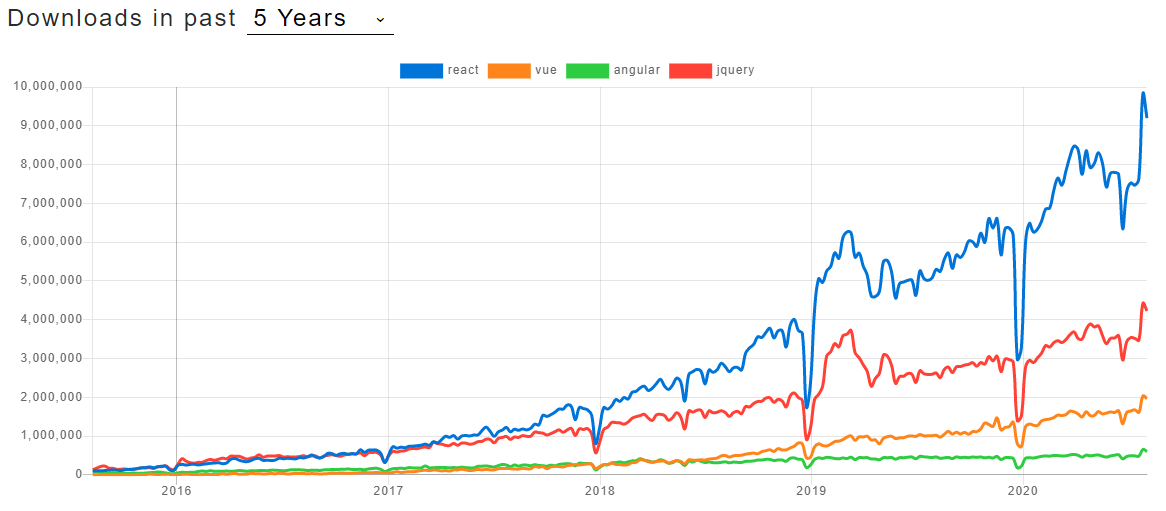


Ilustración 5: Recuento de descargas en NMP trends

<https://www.npmtrends.com/react-vs-vue-vs-angular-vs-jquery>

Características:

* Universalidad: una PWA debe funcionar sin problemas para cada usuario, independientemente de su navegador web.
* Responsive: debe funcionar con cualquier dispositivo, ordenador portátil, tablet o teléfono inteligente.
* Diseño: debe imitar las aplicaciones móviles nativas, es decir, menús simplificados y fáciles de encontrar, con una interactividad simple para funciones avanzadas.
* Seguridad: PWA deben usar siempre HTTPS para mantener seguros los datos del usuario.
* Actualizaciones: los PWA se mantienen actualizados y ofrecen las últimas versiones de un servicio o sitio.
* Descubrimiento: los usuarios deben poder encontrar fácilmente las PWA y ser identificables como una aplicación, en lugar de un sitio web.
* Instalación: desde cualquier navegador sin pasos adicionales y sin necesidad de descargas desde tiendas de aplicaciones.
* Compartir: PWA solo requiere una única URL para compartir, sin ninguna instalación.

[La clave](https://codigofacilito.com/articulos/progressive-apps) de las aplicaciones web progresivas son los llamados ***service workers***. Es un script (un proxy controlable que se ejecuta entre el navegador y el servidor) «separado de una página web, abriendo la puerta a funciones que no necesitan una página o la interacción del usuario». Como tal, estos service workers forman la base del estándar PWA, utilizando caché web para obtener resultados casi instantáneos.

Los service workers **solo duran mientras se requiera su acción**. En una PWA, cuando haces clic en algo o usas una función, uno de ellos entra en acción, procesa el evento decidiendo si la caché sin conexión puede completar la solicitud. La idea es que haya múltiples cachés fuera de línea para que PWA la selecciones, lo que brinda una gama mucho más amplia de funcionalidades sin conexión.

Además, la caché no es solo para aumentos de velocidad fuera de línea. Por ejemplo, si ejecutas una PWA, pero tu conexión es extremadamente irregular el service workers puede servir una caché anterior, en pleno funcionamiento, sin interrumpir la experiencia. De hecho, hay sitios web, tan bien implementados, que cargan de manera rapidísima, **incluso más rápido que una app nativa**.

## Comparativa de las soluciones seleccionadas.

Observando la tendencia de popularidad en la ilustración 6 donde se muestra los datos ofrecidos por DB-Engines, se aprecia como a partir de 2016, mientras que Solr se ha estabilizado en popularidad, Elasticsearch va en aumento; alcanzándolo a partir de esta fecha y superando, año tras año, a esta herramienta; pasando a ser el motor de búsqueda más popular.

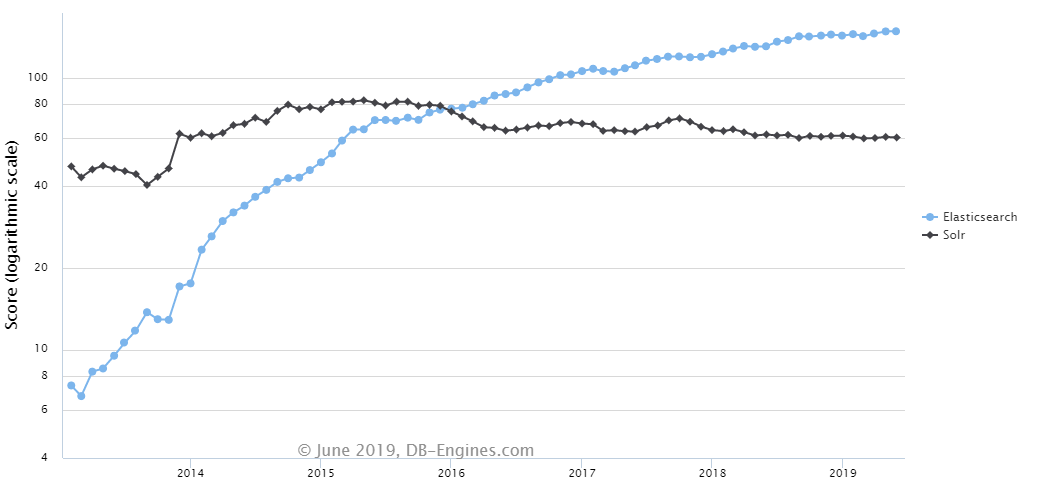


Ilustración 6: DB-Engines ranking Elasticsearch vs Solr

*Fuente: https://db-engines.com/en/ranking\_trend/system/Elasticsearch%3BSolr*

En cuanto a otras características que también son ofrecidas y comparadas por DB-Engine tenemos la siguiente tabla:

|  |  |  |
| --- | --- | --- |
| Nombre | Elasticsearch | Solr |
| Descripción | Un moderno motor de búsqueda y análisis RESTful distribuido basado en Apache Lucene. | Un motor de búsqueda empresarial ampliamente utilizado basado en Apache Lucene. |
| Modelo de base de datos primaria | Motor de búsqueda | Motor de búsqueda |
| Modelos de base de datos secundarios | Basadas en documentos |  |
| Clasificación DB-Engines (tendencia). Puntuación | 148.82 | 60.48 |
| Posición general dentro de los DBMS | 7 | 16 |
| Posición dentro de los DBMS motores de búsqueda | 1 | 3 |
| Sitio web | www.elastic.co/products/elasticsearch | lucene.apache.org/solr |
| Documentación técnica | www.elastic.co/guide/en/elasticsearch/reference/current/index.html | lucene.apache.org/solr/resources.html |
| Desarrollador | Elastic | Apache Software Foundation |
| Versión inicial | 2010 | 2004 |
| Lanzamiento actual | 7.0.0, abril 2019 | 8.1.0, mayo 2019 |
| Licencia | Fuente abierta | Fuente abierta |
| Solo en la nube | no | no |
| Ofertas de DBaaS (enlaces patrocinados) | Servicio de Elasticsearch en Elastic Cloud : pruebe la oferta oficial de Elasticsearch y Kibana disponible en AWS y GCP que es impulsada por los creadores de Elasticsearch. |  |
| Lenguaje de implementación | Java | Java |
| Servidor de sistemas operativos | Todo el sistema operativo con una máquina virtual Java. | Todos los sistemas operativos con una máquina virtual Java y un contenedor de servlets. |
| Esquema de datos | libre de esquemas | sí |
| Mecanografía | sí | sí |
| Soporte XML | no | no |
| Índices secundarios | sí | sí |
| SQL | Lenguaje de consulta tipo SQL | no |
| APIs y otros métodos de acceso | API de Java  RESTful HTTP / JSON API | API de Java  RESTful HTTP API |
| Lenguajes de programación soportados | .Net  Groovy  Java  JavaScript  Perl  PHP  Python  Ruby  Clientes aportados por la comunidad (sin soporte oficial de elasticsearch) | .Net  Erlang  Java  JavaScript  cualquier lenguaje que admita sockets y ya sea XML o JSON  Perl  PHP  Python  Ruby  Scala |
| Scripts del lado del servidor | sí | Plugins de Java |
| Disparadores | sí | sí |
| Métodos de partición | Fragmentación | Fragmentación |
| Métodos de replicación | sí | La nube / distribuye (a través de Zookeeper) la  replicación Maestro-esclavo |
| Mapa reducido | Conector ES-Hadoop | no |
| Conceptos de consistencia | Consistencia eventual | Consistencia eventual |
| Llaves extranjeras | no | no |
| Conceptos de transacción | no | bloqueo optimista |
| Concurrencia | sí | sí |
| Durabilidad | sí | sí |
| Capacidades en memoria | Integración de Memcached y Redis | no |

Tabla 1: Tabla comparativa Elasticsearch vs Solr

*Fuente: https://db-engines.com/en/system/Elasticsearch%3BSolr*

## Acerca de las solucione seleccionada.

#### Solr.

Solr, es la plataforma de búsqueda de código abierto perteneciente al proyecto Apache Lucene ™ dentro de Apache Software Fundation. Está diseñado para impulsar potentes aplicaciones de recuperación de documentos o contenido dondequiera que necesite proporcionar datos a los usuarios en función de sus consultas. No sólo hace uso de la librería de Lucene Core sino que además amplía sus funcionalidades.

Fue Yonik Seely quien creó Solr en 2004 para añadir capacidades de búsqueda al sitio web de la compañía de CNET Networks. En enero de 2006, se realizó un proyecto de código abierto bajo Apache Software Foundation. Hasta la fecha, Solr ha lanzado varias versiones siendo la última, hasta la fecha, el 14 de marzo de 2019 con Apache Solr 8.0.0.

El uso de Solr no sólo contempla las operaciones relacionadas con el manejo de las búsquedas sino también puede utilizarse con fines de almacenamiento; por lo tanto, Al igual que otras bases de datos NoSQL, es una tecnología de procesamiento y almacenamiento de datos no relacionales y búsqueda.

Solr es una capa de sowftare por encima de la API de Java de Lucene. Por lo tanto, aprovecha todas las características que posee Lucene e incluye algunas más. Dentro de las funcionalidades o características que posee Solr se encuentran:

**Capacidades avanzadas de búsqueda de texto completo.** Apoyado en la tecnología de Lucene ™ , Solr permite potentes capacidades de emparejamiento que incluyen frases, tokens, comodines, corrector ortográfico, autocompletado y mucho más en cualquier tipo de datos.

**Interfaces abiertas basadas en estándares - XML, JSON y HTTP.** Para permitir el desarrollo de aplicaciones más sencilla.

**Interfaz de administración.** Solr proporciona una interfaz de usuario administrativa integrada para facilitar el control de sus instancias. Además, desde ella, se permite realizar todas las tareas posibles, como administrar registros, añadir, eliminar, actualizar y buscar documentos.

**Alta escalabilidad y tolerancia a fallos.** Permite escalarse hacia arriba y hacia abajo, realiza una replicación, distribución y balanceo proporcionando tolerancia a fallos. Un escenario relativamente común es que pueden tenerse tantos datos, o tantas consultas, que un solo servidor Solr no puede manejar toda su carga de trabajo. En este caso, puede escalar las capacidades de su aplicación utilizando **[SolrCloud](https://lucene.apache.org/solr/guide/7_7/solrcloud.html" \l "solrcloud)** para distribuir mejor los datos y el procesamiento de solicitudes en muchos servidores. Puede configurarse en múltiples opciones dando distintas combinaciones según la escalabilidad que necesite.

**Monitorización sobre las instancias.** Solr publica sus datos métricos a través de JMX para permitir su monitorización.

**Indexación casi en tiempo real.** Aprovechándose de las capacidades de indexación casi en tiempo real que le proporciona Lucene para ver el contenido inmediatamente después de haberlo indexado.

**Arquitectura Extensible de Plugins.** Solr publica puntos de extensión facilitando la inserción de complementos de indexado y de tiempo de consulta. Además, al ser código abierto facilita la modificación del código en función de las necesidades de las aplicaciones.

**Modo de esquema** o no en función de cómo se necesite. Permite trabajar sin esquema de datos de Solr en entornos no productivos y posteriormente incluirlos en el entorno de producción.

**Extensiones**. Contiene complementos adicionales opcionales para indexar contenido enriquecido como PDF, Word, etc, detección de idioma o agrupación de resultados de búsqueda

**Búsqueda y filtrado de facetas**. Las búsquedas facetadas consisten en obtener listas reducidas de resultados basadas en las propiedades o atributos semánticos de los datos, en lugar de por relevancia. Solr, proporciona este tipo de búsquedas permitiendo fragmentar los datos de la forma más conveniente, usando una gran variedad de algoritmos de facetado

**Búsqueda geoespacial.** Soporta la búsqueda basada en la ubicación.

**Análisis de texto configurable**. Tiene soporte para la mayoría de los idiomas más hablados en el mundo (inglés, chino, japonés, alemán, francés, español y muchos más) y muchas otras herramientas de análisis diseñadas para que la indexación y consulta de su contenido sea lo más flexible posible.

**Cache altamente configurable y extensible por el usuario.** Controles para el cache de información que facilitan la optimización del rendimiento.

**Seguridad.** Seguridad haciendo uso del protocolo SSL, autenticación y autorización basada en roles.

**Opciones avanzadas de almacenamiento**. Basándose en las capacidades de almacenamiento avanzadas de Lucene (códecs, directorios y más), hace que sea más fácil ajustar sus necesidades de almacenamiento de datos para que se ajusten a su aplicación.

**Sugerencias de consulta, ortografía y más**. Proporciona capacidades avanzadas para autocompletar (búsqueda tipográfica), corrección ortográfica, etc.

**Análisis de documentos enriquecidos.** Solr junto con Apache Tika, facilita la indexación de contenido enriquecido, como Adobe PDF, Microsoft Word, etc.

**Índices de búsqueda múltiple**. Es compatible con arquitecturas multi-tenant (multi-cliente) facilitando el aislamiento de usuarios y contenido.

Solr puede ejecutarse en modo servidor único y en modo Cloud.

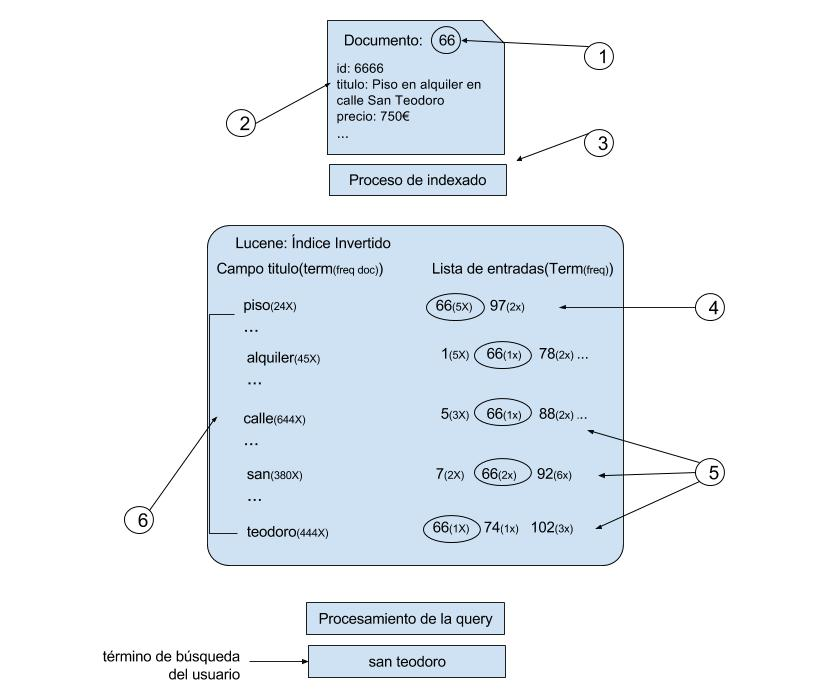
| **SOLR** | **SOLRCloud** |
| --- | --- |
| Modo un solo servidor. Indice == Core. Configuración directamente sobre los archivos de configuración del core. Recomendado para índices pequeños y donde la escalabilidad / tolerancia a fallos no es un requerimiento. | Modo distribuido (búsquedas e índices). Requiere un ensemble de Zookeeper. Requiere un cluster de SOLR. Indice == Collection -> Distribuido en Shards Los shards tienen réplicas Configuración y administración de los shards a través de Zookeeper. No se hacen modificaciones directamente en SOLR. APIS para el manejo de configuración, colecciones, y shards. |

Fuente: https://blog.irontec.com/apache-solr-tutorial/

Solr hoy en día es utilizado en empresas como Panasonic, EBay o Netflix.

El proceso de forma simple sería, partiendo de la indexación de los documentos que queremos tratar para poder realizar su búsqueda.

* Aplicar un identificador interno al documento
* Indicarle unos campos y su valor
* Por cada campo, se analiza e identifican los términos únicos y las frecuencias para cada documento.
* Almacenar en el índice un listado de las entradas junto con el ID del documento y la frecuencia del término para cada campo del documento
* 5 Para el término de búsqueda: “san teodoro”, los resultados incluirán los documentos 7, 66, 74, 92 y 102 porque contienen “san” y “teodoro”



#### Elasticsearch.

Apoyándose en las librerías de Lucene tenemos también el motor de búsqueda y análisis de texto completo, distribuido y RESTful ElasticSearch; está desarrollado en Java y es de código abierto bajo las condiciones de la licencia Apache. Como prácticamente cualquier producto nace en base a una idea sobre el año 2000. El fundador, Shay Banon, que no estaba trabajando por aquel entonces, crea una primera versión de un motor de búsqueda para ayudar a su mujer a buscar recetas. La primera versión del producto aparece en 2010 con el nombre de Compass. y en 2012 se funda Elastichsearch Inc. Al mismo tiempo, otros dos productos están viendo la luz y en 2012 forman un equipo, apareciendo en el mercado y dando como como resultado el ELK Stack: Elasticsearch, Logstash y Kibana Stack. En 2015 se decide cambiar el nombre a la empresa pasándose se a llamar Elastic representando mejor los tres proyectos. Entre las características que tiene se encuentran:

**Near Real Time (NRT).** Elasticsearch es una plataforma de búsqueda casi o próximo a tiempo real. Existe un ligero retardo o latencia desde el momento en que un documento es indexado hasta el momento en que se puede buscar. (aproximadamente un segundo)

**Escalabilidad horizontal**. El diseño de su arquitectura se centró en soportar esta característica. Se tienen varios nodos trabajando como servidor único pudiendo ampliarse de forma indefinida. De esta manera se independiza del hardware (escalabilidad vertical).

**Tolerancia a fallos en los nodos.** Al tener replicación de datos en nodos diferentes ofrece alta disponibilidad y tolerancia a fallos, es decir, permite seguir funcionando correctamente, aunque haya un fallo en uno o varios nodos. Lo consigue gestionando los ‘shards’ y replicándolos.

**Búsqueda en texto completo.** A diferencia de lo que ocurre en las bases de datos tradicionales donde sólo se consideran ciertos campos concretos para realizar una búsqueda, la búsqueda en texto completo se considera todo el contenido del documento.

**JSON.** Orientado a documentos JSon.

**RESTful Api**. Permite un desarrollo e implementación amigable y sencilla en base a su api para su integración.

**Almacenamiento basado en índices**. Los índices son la unidad principal y se dividen en shards (fragmentos) cada fragmento puede ubicarse en un nodo diferente del clúster.

**Integrado con otras herramientas**. Está integrado con otras herramientas del mismo fabricante como Kibana, para la visualización de informes y Logtash, para la administración de logs.

## Bootstrap.

Los Sistemas de Gestión de Contenidos o en inglés Content Management Systems (CMS) se tratan de una herramienta software con el fin de crear y mantener contenido digital. Estas herramientas están dotadas de una interfaz de administración que permite la gestión y mantenimiento del propio sitio web en sí mismo, como de su contenido, ya sea éste en formato de documento Word, pdf, como html y/o imágenes.

Por su tipología nos podemos encontrar con CMS orientados a:

* Contenidos Empresariales (ECM). Son los que permiten gestionar cualquier tipo de contenido generado en una organización, de manera integral y en relación a sus procesos internos de ésta, desde una factura hasta un contenido Web.
* Contenidos Web (WCM). Herramientas orientadas a crear, editar, modificar y publicar contenidos Web, especialmente a portales Web. La privacidad y uso puede ser público o privado. Habitualmente incluyen funcionalidades avanzadas no específicas de estos sistemas como por ejemplo gestión de permisos, sistemas de búsquedas, entornos colaborativos, etc.
* Documentos y/o contenidos multimedia (DMS). Herramientas orientadas a la administración avanzada de todo tipo de ficheros digitales. Entre las funciones más habituales se encuentran localización, clasificación, indexación, recuperación, control de accesos, control de versiones, seguimiento de cambios, etc.

Aunque los ECM son las soluciones más completas para la gestión de contenidos, en muchas ocasiones los Gestores de Contenidos específicos como los WCM son muchos más adecuados ya que involucran menos recursos (humanos, económicos y de infraestructura). Además, hay que añadir la evolución en los últimos años de los WCM, a nivel cualitativo hablando, ha sido importante añadiendo muchas más funcionalidades, por ejemplo, muchos WCM incluir la gestión de los flujos de trabajo para la publicación de contenidos Web, entornos colaborativos para edición de contenidos o funcionalidades propias de los DMS como las búsquedas de contenidos, etiquetado, el control de versiones, etc. Con lo que

De ahora en adelante, cuando se utilice el acrónimo CMS, se estará haciendo referencia en realidad a Gestores de Contenidos Web (WCM), pues dicha acepción es la más extendida y aceptada en el mercado.

 Estos gestores de contenido están conformados o pueden conformarse con distintos tipos de roles y permisos en función de las competencias definidas para los distintos usuarios, con lo que existirán permisos para poder crear o no contenido, editarlo, gestionarlo y publicarlo. Esto implica que los CMS conllevan un módulo para una gestión de usuarios.

El CMS genera páginas web dinámicas interactuando con el servidor web para generar la página web bajo petición del usuario, con el formato predefinido y el contenido definido para las diferentes páginas.

Las funciones principales que posee un CMS se resumen en:

* Funciones para el Sitio Web
  + La creación de un sitio web.
  + La gestión y mantenimiento del sitio.
  + La administración de sitio web.
* Funciones del contenido
  + Creación de contenido o información. Proceso en el que el usuario con los permisos adecuado puede crear nueva información o contenido en distintas modalidades y formatos (textos, gráficos, audios, etc.) dejándolo puesto a disposición de ser publicado.
  + Presentación del contenido o información.
  + Publicación del contenido o información.
  + Mantenimiento del contenido o la información incluyendo su modificación o eliminación.

### Marco de referencia

Se ha hecho uso de las siguientes fuentes de información para recopilar distinta información sobre la tecnología.

* **W3Techs:**  Organización que se tiene como misión analizar el uso de diferentes tecnologías usadas para el desarrollo y la ejecución de sitios web en internet.
* **HostAdvice**: Fuente de opiniones de usuarios y expertos sobre servicios de alojamiento web. Las opiniones y datos tratan sobre precios, calidad y soporte técnico para servidores compartidos, etc.
* **BuiltWith**: Herramienta web que permite analizar, entre otras, las tecnologías utilizadas por un sitio web, así como las tendencias de mercado para distintas soluciones.
* **Cuadrante Mágico de Gartner**: recogen anualmente una clasificación en base a unos informes generados por la misma compañía sobre distintas soluciones del mercado, ayudando a los responsables de la experiencia digital y del cliente a encontrar las herramientas más adecuadas a sus requisitos.

**W3Techs** sitio web que recopila y procesa información sobre el uso de las tecnologías web mediante la aplicación de varias técnicas. El propio sitio advierte que la información que proporciona puede ser incompleta y/o inexacta dada la cantidad y complejidad del tema y también debido a que gran parte de la información es proporcionada por los propios responsables de los sitios y porque no siempre las herramientas que se utilizan, en ocasiones de forma manual, para el descubrimiento de las tecnologías no es 100% fiable y no siempre se rastrean en totalidad las páginas de un sitio web con lo que pueden existir otras tecnologías detrás de estas páginas y que no estén contabilizadas.

Las encuestas que realizan tienen en cuenta:

* Se investigan tecnologías de sitios web, no de páginas web individuales. Si encontramos una tecnología en cualquiera de las páginas, se considera que la utiliza el sitio web.
* En las estadísticas se incluyen sólo los 10 millones de sitios web principales para limitar el impacto de los spammers de dominio.
* Se hace uso de los rankings de popularidad del sitio web proporcionados por Alexa (compañía de Amazon.com).
* No se tiene en cuenta que los subdominios sean sitios web separados. Por ejemplo, se considera que sub1.example.com y sub2.example.com pertenecen al mismo sitio que example.com.
* No se incluyen en el uso los dominios redirigidos. Por ejemplo, Sun.com redirige a Oracle.com y, por lo tanto, no se cuenta.

La siguiente ilustración proporciona la información que desde 2010 se recoge para los CMS, se observa que la Wordpress, Joomla y Drupal encabezan la lista de los CMS con más uso, el último, Liferay, de este listado ocupa la posición 46 pero w3Techs, hoy en día, evalúa un total de 636 plataformas CMS.

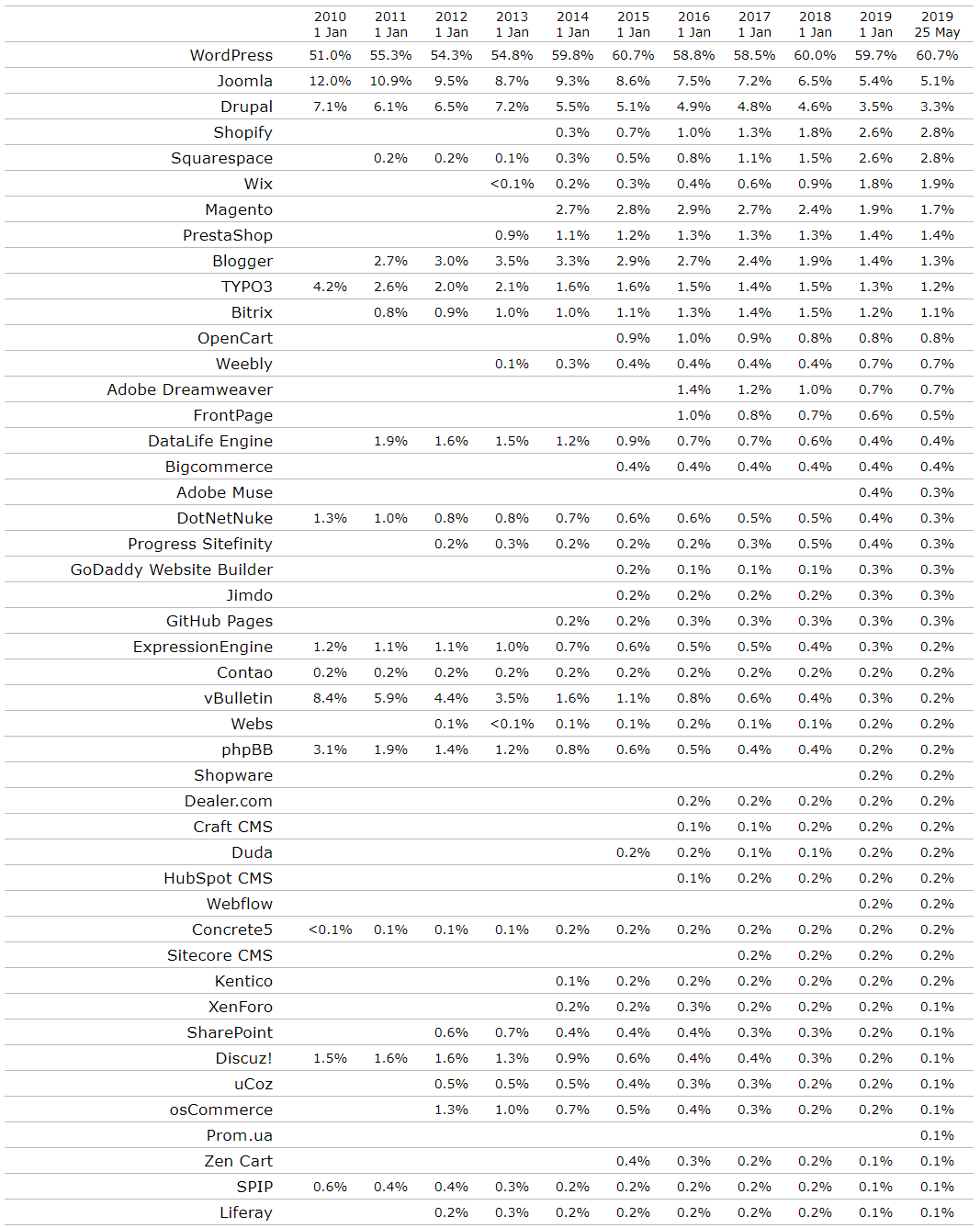


Ilustración 7 : Tendencia histórica en el uso de los gestores de contenidos web desde mayo de 2018

*Fuente: https://w3techs.com/technologies/history\_overview/content\_management/ms/y*

Se ha tenido en cuenta también los datos de mercado de **HostAdvice**. HostAdvice es el centro de información de hospedaje web de Internet, que abarca docenas de empresas y servicios de hospedaje en todo el mundo. Tienen como fin construir una comunidad de de usuarios de alojamiento web para crear la base de datos de servicios de alojamiento más grande y confiable del mundo ayudando a los propietarios de sitios a obtener la mejor relación calidad (o valor) /precio. En base a usuarios expertos, se analizan los mejores servicios de alojamiento web, en base a su confiabilidad, soporte, precios y rendimiento. Así mantienen una base de datos amplia, fácil de usar, que se actualiza con frecuencia y que es accesible para todos. Se puede hacer búsquedas por por tipo de alojamiento (Shared / VPS / Dedicated / Cloud), sistema operativo, métodos de pago, compatibilidad con CMS e incluso ubicación global del servidor.

HostAdvice reúne la información en base la retroalimentación combinada de miles de webmasters de todo el mundo y cada uno de ellos contribuye y hace revisiones de sus propias experiencias personales en el uso de diferentes empresas de hosting para el desarrollo web, soporte para aplicaciones móviles y publicación en la Web. Clasifican a las 50 principales empresas de web hosting basado en que las empresas son recomendadas constantemente por las revisiones de los webmasters, programadores, diseñadores y desarrolladores actuales. Dentro de los CMS realizan una comparativa de mercado de los principales 30.

En la siguiente figura, donde se filtran los datos para realizar la comparativa de 2019 en España aparece en cabeza, de nuevo, WordPress, seguido de Joomla, PrestaShop y Drupal, obteniendo datos muy similares a los ofrecidos por **W3Techs.** En España, sin embargo, se muestra Liferay como séptima posición.

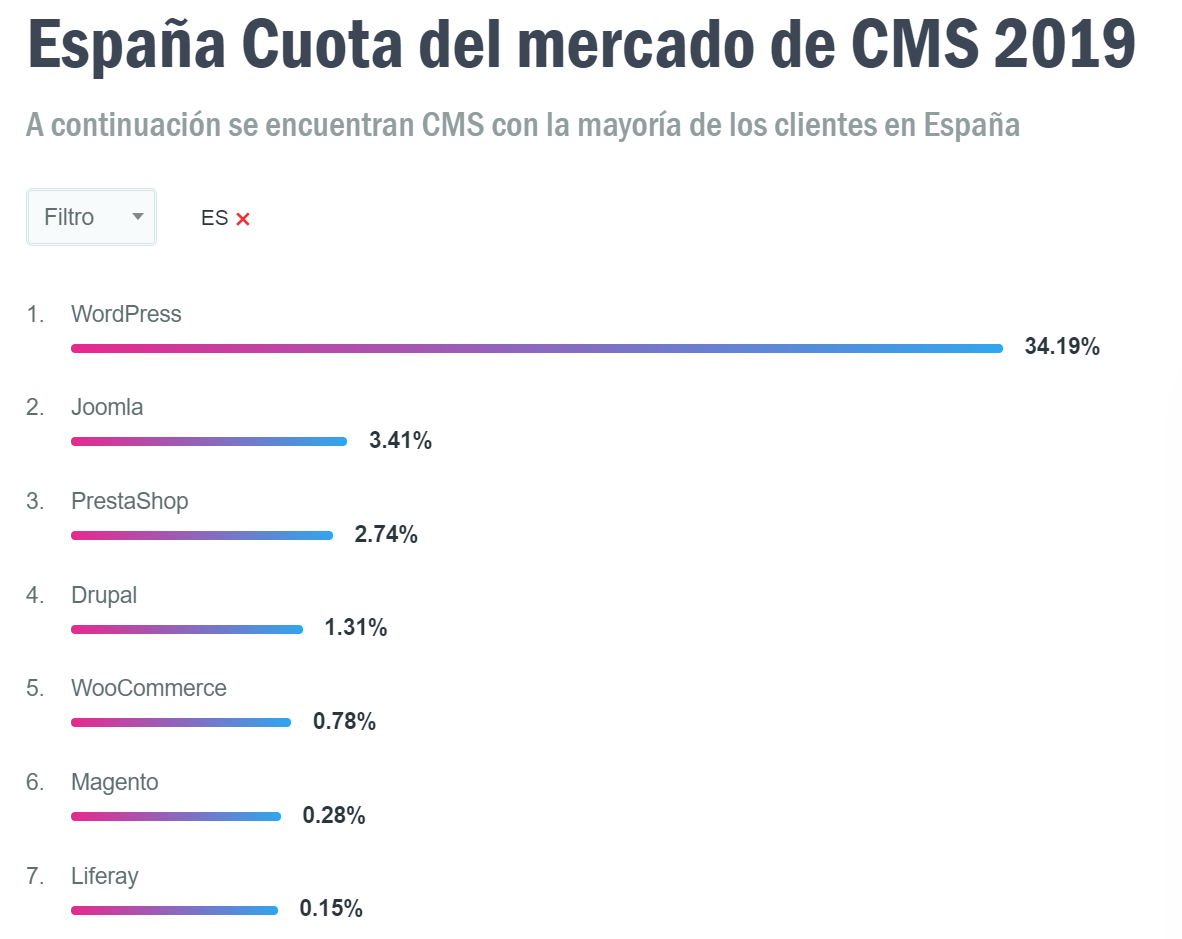


Ilustración 8: Cuota de mercado CMS 2019 en España

*Fuente:* [*https://es.hostadvice.com/marketshare/cms/es/*](https://es.hostadvice.com/marketshare/cms/es/)

BuiltWith®. Fundada en 2007, se trata de una herramienta online que permite averiguar sobre qué tecnología está desarrollado cualquier sitio web. En base a una la dirección web (URL) que se le proporcione, informa, con que CMS está gestionado, que sistemas de publicidad utiliza, los widgets que tiene en funcionamiento, el idioma, el tipo de servidor donde se encuentra alojado, las herramientas utilizadas para el análisis del tráfico o las tecnologías en que está desarrollado, entre otros datos.

Además del análisis individual por sitio, entre sus herramientas se encuentran un analizador de tendencias de tecnología web, listas de palabras clave, listas para eCommerce (categorizando sitios con componentes de comercio electrónico), Acceso mediante API y más.

Para analizar las tendencias sobre las tecnologías web, esta herramienta permite observar estas tendencias en base a un grupo tecnológico, filtrando por país o localización y herramienta concreta. En el caso de España y para seleccionando el grupo tecnológico de los sistemas de gestión de contenido se obtienen los siguientes resultados:

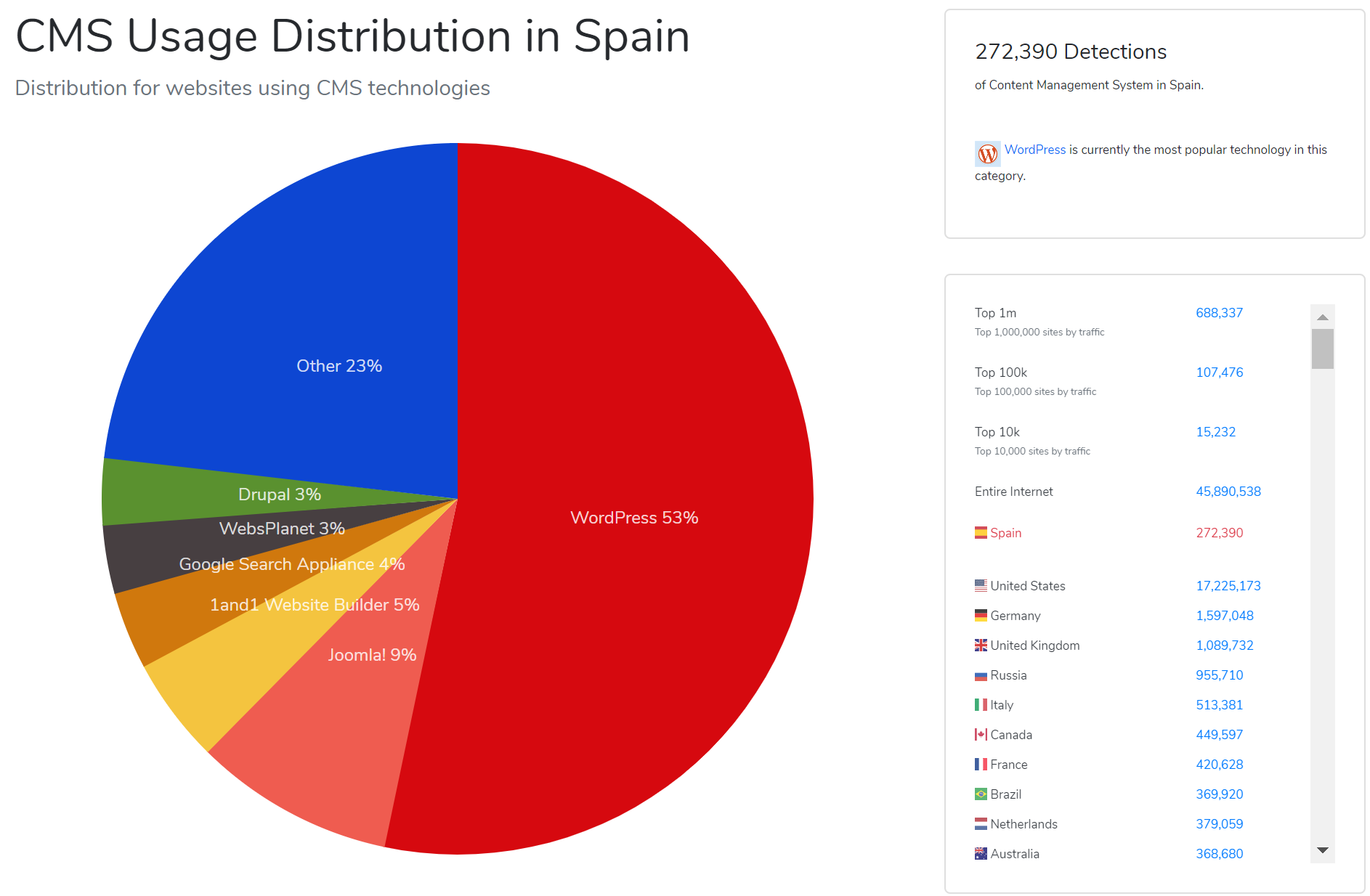


Ilustración 9 – Tendencias de CMS en España

Fuente: https://trends.builtwith.com/cms/country/Spain

WordPress encabezaría la lista con más del 50% de sitios analizados en España que hacen uso de esta herramienta, le seguiría a mucha más distancia Joomla con el 9% y 1 and 1 Website Builder con el 5%.

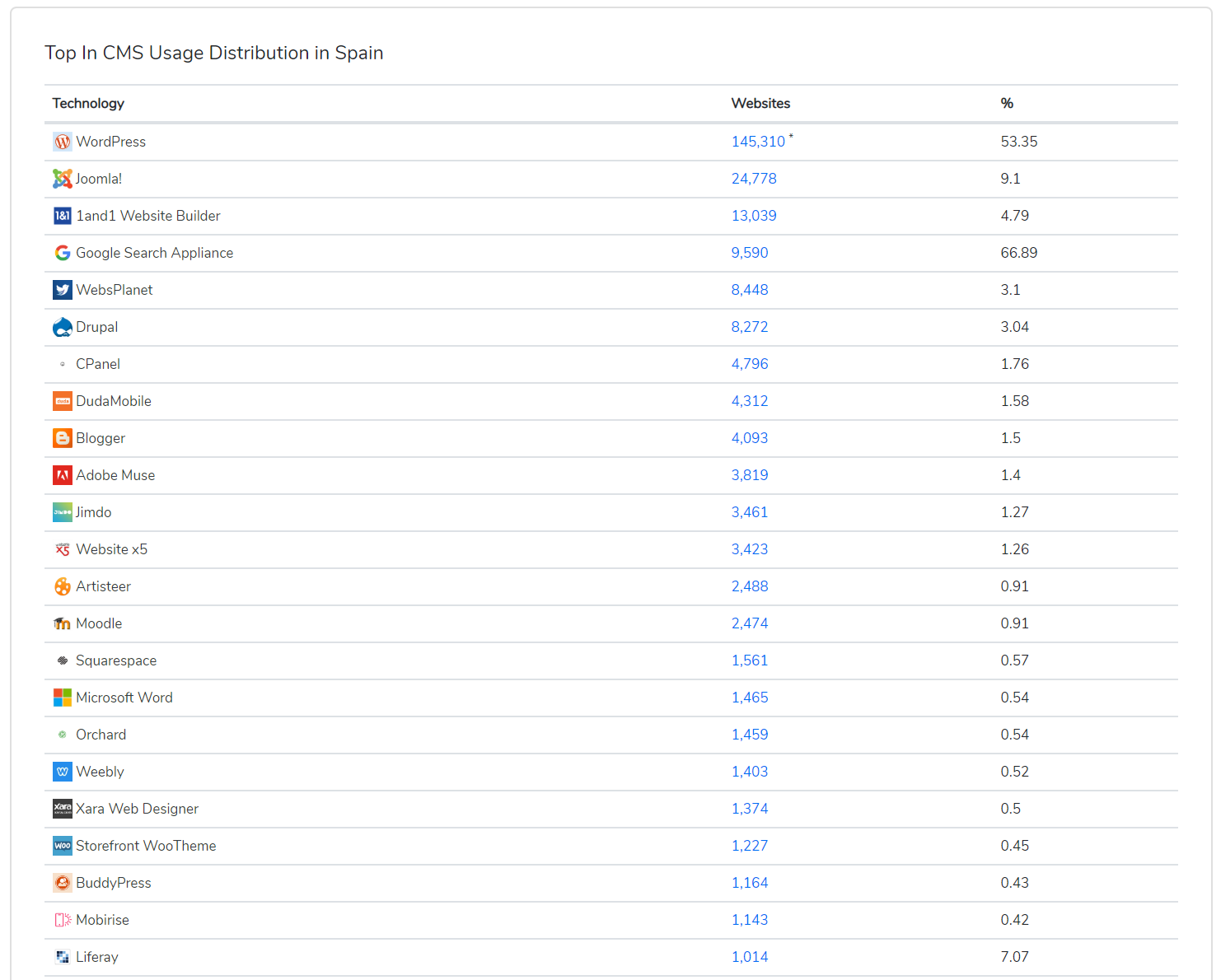


Ilustración 10: Listado de los CMS en España

*Fuente: https://trends.builtwith.com/cms/country/Spain*

Fundada en 1979, **Gartner**, es una empresa de análisis (para brindar consejo a los profesionales de las tecnologías de la información y las comucicaciones), investigación tecnológica (por mercado, por sectores y por áreas) y consultoría.

Los Cuadrantes Magicos, dan un posicionamiento gráfico competitivo dividido en cuatro tipos de proveedores de tecnología en función de dos ejes, capacidad de ejecución y amplitud de visión:

* **Los líderes**; buena capacidad de ejecución y buena visión actual para el futuro.
* **Los visionarios** entienden a dónde va el mercado, pero aún no tienen buena capacidad de ejecución.
* **Los jugadores especializados** se enfocan segmento pequeño y no superan o innovan a otros.
* **Los retadores** buena capacidad de ejecución y pueden dominar un segmento amplio pero no demuestran una comprensión de hacia donde se dirige el mercado.

Según el cuadrante mágico de Gartner relativo a las plataformas de experiencia digital (DXP), estas poseen ciertas capacidades que pueden definirse en las siguientes tres secciones principales de los criterios funcionales para este mercado:

|  |  |  |
| --- | --- | --- |
| Experiencia del cliente | Gestión de DXP | Plataforma / arquitectura |
| Interacción de contenido  Búsqueda, navegación y descubrimiento.  Colaboración y gestión del conocimiento.  Experiencia de personalización  Comercio digital | Gestión de contenido  Integración y agregación  Personalización y focalización.  Analítica y optimización  Administracion de seguridad  Flujo de trabajo / gestión de procesos de negocio.  Desarrollo | Entrega y presentación multicanal.  Gestión de datos de clientes  Habilitación de la nube  Globalización / localización / soporte multilingüe |

Según el cuadrante mágico para plataformas de experiencia digital presentado este último año en febrero, posiciona a Liferay como la mejor solución.



Ilustración 11: Cuadrante Mágico de Gartner 2019

Fuente: https://www.gartner.com/doc/reprints?id=1-67R0V6S&ct=190211&st=sb

### Soluciones seleccionadas.

Según se ha visto en el apartado anterior, existen cientos de soluciones disponibles en el mercado para la gestión de contenidos web. El objetivo de este trabajo no es analizarlos todos y cada uno de ellos sino en base a unos criterios, seleccionar el que más se adapte a los requisitos y necesidades planteadas para satisfacer al “CLIENTE”, así como el escenario donde tiene que ser implantado.

Para realizar la selección final de los 3 CMS sobre los que realizar la comparativa, se ha tenido en cuenta un conjunto diverso de características, las más relevantes han sido: licencia de uso (software libre y código abierto), posicionamiento o cuota en el mercado y valoración por el “CLIENTE”, así pues los CMS finalmente seleccionados han sido:

* Wordpress
* Drupal
* Liferay

Aunque Liferay no tiene tan buena tendencia de mercado como arroja el marco de referencia revisado, se incluye a petición del “CLIENTE” puesto que es conocedor de la herramienta y se hace uso de ella para otros organismos de la administración pública valenciana.

### Comparativa de las soluciones seleccionadas.

Aunque no se ha hecho uso del proyecto CMS Matrix para analizar el marco de referencia ya que los datos que figuran en él están actualizados sólo hasta 2012, sí ha servido como base para realizar la comparativa de los tres CMS seleccionado.

CMS Matrix permite comparar fácilmente un conjunto de características entre más de 1300 CMS del mercado. La información ha sido recogida en base a una comunidad de usuarios abierta, con los riesgos que conlleva. Para realizar la comparativa sólo hay que marcar los productos/herramientas sobre los que se está interesado y al comparar se muestra, en forma de tabla, un conjunto de características fijas agrupadas por áreas y por cada una de las herramientas el detalle para característica.

Las características están agrupadas en:

* **Requerimientos del sistema**: Conjunto de requerimientos necesarios que indica el fabricante para su correcto funcionamiento.
* **Seguridad**: características relativas a la seguridad de la herramienta y funcionalidades que implementa.
* **Soporte**: Servicios que ofrece que el fabricante para ayudar a los usuarios más o menos técnicos ante dudas, sugerencias, problemas, ect.
* **Usabilidad**: Funcionalidades que facilitan tareas a los usuarios que gestionan contenido del sitio.
* **Rendimiento**: Se indica en esta área el conjunto de características relacionadas con el rendimiento del CMS.
* **Administración**: Conjunto de funcionalidades y características que ayudan la administración del CMS.
* **Interoperabilidad**: Grupo de características relativas a habilidad de dos o más sistemas/componentes para intercambiar información y la utilización de la información intercambiada
* **Flexibilidad**: Características que ayudan y flexibilizan la realización de las tareas.
* **Funcionalidades-Extra**: Características que ayudan y dan funcionalidad extra al sitio.

A continuación, se muestran por cada un de las soluciones los resultados a cada una de las características que han sido valoradas e interesantes desde el punto de vista del cliente.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Wordpress | Druppal | Liferay CE Portal |
| Version | 5.2.1 | 8.7.2 | 7.1 |
| System Requirements | | | |
| Application Server | Apache/Nginx | Cualquiera con soporte PHP | Tomcat/Wildfly |
| Approximate Cost | Gratuito | Gratuito | Gratuito (disponible versión Edición Empresarial) |
| Database | MariaDB/MySQL | MySQL | MySQL |
| License | GNU/LGPL (libre y código abierto) | GNU/LGPL (libre y código abierto) | GNU/LGPL (libre y código abierto) |
| Programming Language | PHP | PHP | JAVA |
| Web Server | Cualquiera |  | Cualquiera |
| Security | | | |
| LDAP Authentication | A través de plugin gratuito | SI | SI |
| SSL Compatible | SI | SI | SI |
| SSL Logins | A través de plugin gratuito | SI | SI |
| SSL Pages | A través de plugin gratuito | SI | SI |
| Versioning | SI | SI | SI |
| Support | | | |
| Certification Program | NO | NO | SI |
| Online Help | SI | SI | SI |
| Professional Services | SI | SI | SI |
| Public Forum | SI | SI | SI |
| Test Framework | SI | SI | SI |
| Third-Party Developers | SI | SI | SI |
| Community | SI | SI | SI |
| Ease of Use | | | |
| Drag-N-Drop Content | A través de plugin gratuito | SI | SI |
| Friendly URLs | SI | SI | SI |
| Image Resizing | A través de plugin gratuito | SI | SI |
| Macro Language | A través de plugin gratuito | A través de plugin gratuito | SI |
| Server Page Language | SI | SI | SI |
| Zip Archives | NO | A través de plugin gratuito | SI |
| Performance | | | |
| Advanced Caching | A través de plugin gratuito | SI | SI |
| Database Replication | A través de plugin gratuito | SI | SI |
| Load Balancing | SI | SI | SI |
| Page Caching | A través de plugin gratuito | SI | SI |
| Static Content Export | A través de plugin gratuito | A través de plugin gratuito | SI |
| Management | | | |
| Advertising Management | A través de plugin gratuito | A través de plugin gratuito | SI |
| Asset Management | SI | SI | SI |
| Content Scheduling | A través de plugin gratuito | SI | SI |
| Content Staging | A través de plugin gratuito | SI | SI |
| Inline Administration | SI | SI | SI |
| Online Administration | SI | SI | SI |
| Package Deployment | A través de plugin de pago | A través de plugin gratuito | SI |
| Sub-sites / Roots | SI | SI | SI |
| Themes / Skins | SI | SI | SI |
| Interoperability | | | |
| FTP Support | No permitido | A través de plugin gratuito | NO |
| UTF-8 Support | SI | SI | SI |
| WebDAV Support | NO | NO | SI |
| Flexibility | | | |
| Extensible User Profiles | A través de plugin gratuito | SI | SI |
| Interface Localization | SI | SI | SI |
| Metadata | SI | SI | SI |
| Multi-lingual Content | A través de plugin gratuito | SI | SI |
| Built-in Applications | | | |
| Blog | SI | SI | SI |
| Chat | A través de plugin gratuito | A través de plugin gratuito | SI |
| Contact Management | A través de plugin gratuito | A través de plugin gratuito | SI |
| Data Entry | A través de plugin gratuito | A través de plugin gratuito | SI |
| Database Reports | NO | NO | NO |
| Discussion / Forum | A través de plugin gratuito | SI | SI |
| Document Management | A través de plugin gratuito | A través de plugin gratuito | SI |
| Events Calendar | A través de plugin gratuito | A través de plugin gratuito | SI |
| Events Management | A través de plugin gratuito | A través de plugin gratuito | SI |
| Product Management | A través de plugin gratuito | | SI |
| Project Tracking | A través de plugin gratuito | A través de plugin gratuito | SI |
| Search Engine | SI | SI | SI |
| Site Map | A través de plugin gratuito | A través de plugin gratuito | SI |
| User Contributions | A través de plugin gratuito | A través de plugin gratuito | SI |
| Wiki | A través de plugin gratuito | A través de plugin gratuito | SI |

### Acerca de las soluciones seleccionadas.

#### ****Liferay****

Aunque según las tendencias del mercado y los datos recogidos en el marco de referencia, parece que ha perdido relevancia en este grupo de tecnologías web con respecto a otras herramientas colaborativas y sistemas gestores de contenido, Liferay, permite una gran extensión de funcionalidad a los desarrollos con gran poder de integración con otras herramientas.

Principales características de Liferay:

* Liferay tiene una interfaz de programación fácil de usar que permite a los usuarios crear sitios web básicos y facilita la admiración fácil incluso sin conocimientos de programación. Tiene un sistema de gestión de contenido web integrado que facilita la creación de sitios web y portales empresariales como un conjunto de páginas. Portales, temas así como la navegación común.
* Ofrece soporte para complementos para múltiples lenguajes de programación, incluidos los portlets de Ruby y PHP. Liferay sigue estándares abiertos para contenido, servicios web, tecnologías front-end, portlets, etc. que ayudan a reducir el costo total del desarrollo y ofrecen un retorno de la inversión inmediato.
* Liferay Portal permite a los usuarios, desarrolladores y administradores integrar contenido y servicios desde aplicaciones de back-end o heredadas al admitir múltiples métodos de integración como REST, SOAP, RSS, etc., junto con API propias.
* Debido a su arquitectura liviana, escalable y de código abierto, ayuda a los desarrolladores a cumplir con la alta disponibilidad y las demandas de alto rendimiento de la empresa actual. También permite a los desarrolladores escalar el sistema de manera eficiente a medida que aumenta la base de usuarios.
* Ofrece un modelo de complemento y extensión que permite a los desarrolladores adaptar el comportamiento del producto para que se adapte a sus necesidades sin necesidad de actualizar o reescribir el código desde cero.
* Ofrece una amplia selección de bases de datos, sistemas operativos y servidores de aplicaciones para ejecutar. Esto asegura que la inversión en infraestructura y habilidades esté sucediendo de manera óptima.
* Su funcionalidad altamente modular ayuda a que los sitios web sean más pequeños y, por lo tanto, más eficientes con una mayor velocidad de operación.

**WordPress**

WordPress contribuye al 60% del uso de CMS en todo el mundo. Es uno de los sistemas de blogs de código abierto y CMS más importantes, si no es así, y se ha convertido en una parte integral de la arquitectura de Internet. WordPress potencia el 27% de toda Internet y WordPress 4.6 se ha descargado 21.7 millones de veces. Ahora, estas son figuras convincentes!

Aquí hay un vistazo a algunas características potentes de WordPress:

* Permite a los usuarios desarrollar sitios web potentes y con muchas funciones sin conocimientos técnicos avanzados. Cuenta con una interfaz de usuario muy intuitiva y fácil de usar que permite a los usuarios navegar y usar la plataforma fácilmente.
* Tiene un menor costo de instalación, mantenimiento y personalización, y su instalación, implementación y actualización son gratuitas.
* WordPress ofrece una gran capacidad de extensión al ofrecer más de 47,000 complementos para todos los medios y propósitos. También tiene extensiones generadas por el usuario que aseguran que los sitios web no estén limitados por extensiones creadas solo por la empresa.
* Tiene capacidades para múltiples usuarios y permite que múltiples usuarios contribuyan con contenido web
* Los temas de WordPress son compatibles con dispositivos móviles y receptivos, por lo que pueden adaptarse a cualquier factor de forma. El panel de control de WordPress puede funcionar bien en un teléfono inteligente y en una PC, lo que facilita la administración del sitio web.
* Está optimizado para la velocidad y también ofrece integración de redes sociales.

**Drupal**

Drupal es una plataforma de gestión de contenido web de código abierto que ayuda a los desarrolladores a crear experiencias digitales sorprendentes. Es amigable para el desarrollador y puede utilizarse para diseñar sitios web complejos. Admite algunos de los sitios más concurridos del mundo y es capaz de manejar grandes volúmenes de usuarios y tráfico.

* Drupal ofrece control avanzado de URL y cada elemento de contenido puede recibir una URL individual. Drupal también facilita a los desarrolladores la automatización de la estructura de URL personalizada para diferentes tipos de contenido.
* Cuenta con un poderoso sistema de taxonomía que facilita el etiquetado y la organización del contenido.
* Drupal ofrece una sólida función de usuario y funcionalidad de control de acceso que permite a los desarrolladores crear roles de usuario personalizados ilimitados con niveles de acceso personalizados según sea necesario.
* Es muy flexible y se puede usar para desarrollar sitios web empresariales complejos, blogs simples e incluso sitios web de comercio electrónico de alto tráfico con facilidad.
* En comparación con WordPress, los sitios web de Drupal tienen un tiempo de respuesta más rápido y se cargan fácilmente.
* Ofrece capacidades avanzadas de modelado e integración de datos con una variedad de aplicaciones y servicios.

## Firebase.

Todas las soluciones revisadas en los puntos anteriores tanto en lo que se refiere a de búsqueda como a CMS cumplen las expectativas que se requieren para la solución a desarrollar e implementar, con lo que no puede concluirse que haya ninguna opción claramente ganadora. Cada uno tiene ventajas/desventajas que hace compleja una decisión sobre cual decantarse. Finalmente, y en última instancia, la mejor opción va a ser la que menos riesgos plantee y la más valorada por el “CLIENTE”.

## Conclusiones.

Todas las soluciones revisadas en los puntos anteriores tanto en lo que se refiere a de búsqueda como a CMS cumplen las expectativas que se requieren para la solución a desarrollar e implementar, con lo que no puede concluirse que haya ninguna opción claramente ganadora. Cada uno tiene ventajas/desventajas que hace compleja una decisión sobre cual decantarse. Finalmente, y en última instancia, la mejor opción va a ser la que menos riesgos plantee y la más valorada por el “CLIENTE”.

# Gestión del proyecto.

Entregables y tareas según la fase del proyecto, se indica a alto nivel, una hoja de ruta con la planificación estimada (esta planificación podría sufrir cambios).

Fase de inicio:

* Constitución del proyecto. 29/01/2020

Fase de planificación: En esta fase se realizará la toma de requerimientos globales. Se definirán e identificarán los roles del proyecto así como el recurso asignado a cada uno. Se realizara una aproximación inicial a la arquitectura de desarrollo y del producto final y se establecerá el presupuesto o coste del proyecto.

* **Definir el alcance del proyecto.** El alcance del proyecto todavía no está definido y puede ser variable en función del avance del mismo por este motivo se decide ir hacia metodología agil en gestión de proyectos y un desarrollo del proyecto de forma iterativa.
* **Planificación de las iteraciones.** Las iteraciones se llevarán en periodos de 3 semanas planificandose las siguientes:
  + 29/01/2020
  + 19/02/2020
  + 11/03/2020
  + 01/04/2020
  + 22/04/2020
  + 13/05/2020
  + 03/06/2020
* **Estimar los costes** o presupuesto inicial del proyecto.
* Aun basandose en metodología agil y que en uno de sus 4 valores indica que se valora más el software que funciona que la documentación exhaustiva, sí se detallarán y acordarán los documentos que serán necesarios en las entregas de cada sprint y el nivel de detalle.

Fase de desarrollo (ejecución, monitorización y control del proyecto). En base a las distintas iteraciones o sprints planificados cada 3 semanas se realizará para cada uno de ellos las siguientes

* **Generación del product backlog (proceso continuo).** Lista de características refinadas que son ordenadas, detalladas y estimadas (también podrían ser las historias de usuario transmitidas por el cliente). Se trata de un proceso continuo y vivo. Estas caracteristicas se priorizarán en función de lo que aporte mayor valor al producto.
* **Reunión de planificación de Sprint (0,5 jornadas).** Se recogerá del Product backlog aquellas caracterísitcas que incrementen la funcionalidad al producto generando un Sprint backlog. Se identificarán los riesgos asociados/derivados y se realizará su análisis. Aquí, se definirá el incremento la funcionalidad que va a ser entregado en el producto y el objetivo del sprint así como la forma en la que va a realizarse el trabajo.
  + Requisitos.
  + Diseño técnico .
  + Diseño funcional (casos de uso)
  + Interfaces de usuario.
  + Modelo de datos (conceptual/lógico)
  + Perfiles/roles del producto.
* **Daily (10 minutos diarios).** Se tiene como objetivo el evaluar el progreso del trabajo del sprint así como el control y seguimiento de riesgos.
* **Trabajo de desarrollo. (13,5 jornadas).** En función del conjunto de las características a desarrollar en el sprint se realizarán las siguientes acciones.
  + Construcción/Implementación de las caracteristicas.
  + Modelo de datos (físico)
  + Pruebas unitarios/integración
* **Revisión del Sprint (0,5 jornadas).** Reunión donde se inpecciona el incremento de funcionalidad entregado en el producto y, si es necesario, se adapta el Product Backlog (revisado). Con la presentación del incremento obtenfreomos la retroalimentación necesaria asegurando la calidad del producto y adpatarnos a los posibles cambios de alcance del producto. Además recogeremos las lecciones aprendidas del sprint, los riestos y/o problemas que se hayan materializado y su solución.
* **Retrospectiva del Sprint (0,5 jornadas).** Se revisará el último sprint en lo que respecta a las personas, relaciones, procesos y herramientas, identificando y ordenando los temas que han surgido y las mejoras relativas a como el equipo hace el trabajo para ponerlas en marcha. Mejora continua en el modo de trabajo.

Fase de cierre:

* **Entrega del producto, acta de cierre del proyecto** con el cliente y finalización del mismo 03/06/2020

Dentro de la gestión del proyecto se van a abordar una serie de fases representadas en la siguiente ilustración:

Ilustración 12: Fases de la gestión del proyecto

## Requisitos.

En este apartado se describen las especificaciones trasladadas por el “Cliente” tras varias sesiones y reuniones de trabajo conjuntas. A continuación, se listan y detallan los requisitos solicitados por el “CLIENTE” a la solución analizada y a implementar. De este modo en la fase de desarrollo e implementación estos requisitos servirán de base para asegurar que la solución cumple con lo demandado. Así mismo, se recogen para aquellos requisitos, lo que sale del alcance del trabajo, que puede ser analizado para futuros trabajos de implementación.

## Acceso.

Deberá existir un acceso público, para los usuarios finales que no se hayan autenticado en el portal y otro acceso privado para uso exclusivo del personal interno de la organización. Aunque la autenticación de usuarios dentro de Liferay puede realizarse contra un Directorio Activo del que dispone la organización, se excede del alcance del presente trabajo y sólo existirán usuarios anónimos y los autenticados, cuyo acceso se realizará de forma local.

## Usuarios.

El portal hará diferencia entre usuarios anónimos (no autenticados) que tendrán disponible las funcionalidades públicas y los usuarios locales, que serán aquellos usuarios que hayan sido autenticados de forma satisfactoria en el portal. Dentro de estos usuarios se distinguirán:

* Administradores. Serán los usuarios con capacidad de administrar la herramienta y configurarla.
* Publicadores. Serán aquellos usuarios que podrán publicar los contenidos y documentos que estarán disponibles para los usuarios anónimos.

Aunque la herramienta permite infinidad de roles y permisos, en los requisitos solicitados relativos a los usuarios, sólo se tendrán en cuenta los mencionados ya que el objetivo del trabajo es la implementación de los componentes que forman parte del SRI y no así de otras funcionalidades.

## Búsquedas.

Las búsquedas serán accesibles para todos los usuarios, tanto anónimos como autenticados, se permitirá realizar búsquedas de los documentos por cualquiera de sus campos. Las búsquedas serán auto recursivas, por tanto, no es necesario el campo concreto a buscar, con un fragmento será suficiente para obtener resultados. Se pueden analizar los datos mediante la inserción de texto en la barra del buscador o mediante una ayuda que se despliega al pulsar en el botón de configuración situado junto al buscador y que permite buscar considerando la coincidencia de todos los parámetros completados o solo de alguno.

## Resultado de las búsquedas.

Al igual que en el caso de la búsqueda, esta funcionalidad estará disponible para cualquier usuario una vez se haya procedido a realizar una búsqueda. Así pues, el sistema en base al criterio de búsqueda definido por el usuario, mostrará un listado con los documentos coincidentes a ese criterio de búsqueda.

Este listado contendrá una previsualización del documento.

Además, se permitirá una ordenación del listado por fecha.

## Previsualizar documento.

El sistema debe permitir realizar una previsualización del documento resultado de la búsqueda, sin necesidad de ser descarga del mismo.

## Descargar documentos.

El sistema permitirá a los usuarios la descarga local de aquellos documentos que hayan aparecido en los resultados de las búsquedas.

## Publicación de documentos.

Sólo los usuarios con permisos de ‘Publicador’ podrá gestionar documentos y publicarlos. La publicación de contenidos consistirá:

* Almacenamiento de los contenidos en el portal.
* Definición de las palabras clave.
* Serán admitidos formatos de tipo pdf. Aunque el sistema permite mayor cantidad de formatos, la publicación debe restringirse para este tipo de documentos.
* Deberá permitir versionados en los documentos.
* El sistema debe
* Todos los tipos de contenido se podrán crear y editar mediante un formulario. Para futuros trabajos, sería interesante ofrecer la posibilidad de crear nuevos tipos de contenidos mediante la carga de un fichero de Excel o de documentos adicionales.

## Análisis.

Una vez definidos los requisitos necesarios por el “CLIENTE” se hará una valoración de las alternativas a los productos a utilizar, un análisis del sistema, se detallarán los casos de uso, se realizará un desglose de las tareas a realizar y su estimación, los costes que acarreará la solución a aplicar y los riesgos detectados.

## Análisis de las alternativas y selección.

El sistema permitirá a los usuarios la descarga local de aquellos documentos que hayan aparecido en los resultados de las búsquedas.

## Casos de uso.

En el siguiente apartado se va a proceder a definir los casos de uso los cuales se encargan de describir las interacciones entre la aplicación y los agentes externos, con el fin de obtener los requisitos de la aplicación. Expresan el punto de vista del usuario sobre cómo debe funcionar la aplicación.

Los casos de uso se van a detallar en formato de ficha indicando los siguientes datos:

* **Identificación:** Identificador único del caso de uso.
* **Nombre:** Nombre del caso de uso a tratar.
* **Descripción:** Breve descripción del caso de uso.
* **Casos de uso asociados**: Relación con otros casos de uso.
* **Actores**: Usuarios que participan en el caso de uso.
* **Precondiciones:** Requisitos previos requeridos.
* **Activación:** Modo de activación del caso de uso.
* **Post-condiciones:** Resultados de la realización del caso de uso.
* **Flujo alternativo:** Mencionar el flujo alternativo si lo hubiera.
* **Factores de decisión**: Nivel de frecuencia, criticidad, riesgos, complejidad valorados como ‘Baja’, ‘Media’, ‘Alta’

Si incluye, después de una de las fichas del caso de uso detallado, un diagrama de actividad, que contiene el esquema gráfico para modelar el proceso del caso de uso y listar sus pasos.

A continuación, se detallan los casos de uso en relación con cada uno de los actores intervinientes.

Imagen que contiene texto, mapa

Descripción generada automáticamente

Ilustración 13: Casos de uso por actor

## Acceder al sistema.

#### Ficha

|  |  |  |
| --- | --- | --- |
| **FICHA DE CASO DE USO** | |  |
| **IDENTIFICACIÓN** | CA-001 | |
| **NOMBRE** | Acceder al portal | |
| **DESCRIPCIÓN** | El usuario introduce la dirección del portal en el navegador y accede al sistema. | |
| **CASOS DE USO ASOCIADOS** | * Buscar documentos * Iniciar sesión | |
| **FLUJO NORMAL** | | |
| **ACTORES** | Usuario | |
| **PRECONDICIONES** |  | |
| **ACTIVACIÓN** | Este caso de uso es invocado por cualquier usuario que quiera acceder al portal | |
| **DESCRIPCIÓN** | Se introduce en el navegador la dirección del portal  Se muestra al usuario las opciones de la página de bienvenida | |
| **POSTCONDICIONES** | N/A | |
| **FLUJO ALTERNATIVO** | | |
| **DESCRIPCIÓN** | N/A | |
| **FACTORES DE DECISIÓN** | | |
| **FRECUENCIA** | Media | |
| **CRITICIDAD** | Media | |
| **RIESGOS** | Baja | |
| **COMPLEJIDAD** | Baja | |

#### Diagrama de Actividad

NA

## Buscar documento.

## Mostrar resultados.

## Abrir documento.

## Iniciar sesión.

## Publicar documentos.

## Indexar documentos.

## Layouts.

En este punto van a incluirse los distintos esquemas de distribución de los elementos en las páginas. Para llegar a estos layouts finales, se realizaron varias propuestas al “CLIENTE” siendo los siguientes los seleccionados por el mismo. Estos esquemas sirven de base al desarrollador que tiene que realizar las páginas necesarias para mostrar las funcionalidades requeridas.

## Estimación temporal.

## Costes.

## Costes de material y software.

Al tratarse de herramientas tipo open source y versiones de herramientas de mercado software libre se abarata mucho el coste de los trabajos. Se detallan a continuación los costes en función del material y el software sobre el que se implementa la solución.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Recurso | Coste total | Coste mensual | Tiempo | Total |
| Servidor  Disco duro |  |  |  |  |
| Monitor |  |  |  |  |
| Equipo de sobremesa |  |  |  |  |
| Conexión a internet |  |  |  |  |
| Dominio |  |  |  |  |
| Microsoft Project |  | 3$ | 6 meses | 9$ |
| Herramientas para la solución (software libre)  Liferay CE  Tomcat 9  MySQL  Elasticsearch | 0€ |  |  |  |
| Herramientas de desarrollo (software libre)  Eclipse | 0€ |  |  |  |
| TOTAL |  |  |  |  |

## Costes de recursos humanos.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Concepto/Tarea | Rol del trabajador | Coste/hora (€) | Nº horas | Total € |
| Reuniones | Analista |  |  |  |
| Estudio de viabilidad | Analista |  |  |  |
| Análisis del proyecto | Analista |  |  |  |
| Diseño del proyecto | Analista |  |  |  |
| Construcción del proyecto | Desarrollador |  |  |  |
|  |  |  |  |  |
|  |  | 0€ |  |  |
| Documentación del proyecto | Analista |  |  |  |
| TOTAL | | | |  |

## Riesgos.

En todos los proyectos debe realizarse una gestión de los riesgos que incluyen entre otras cosas, la identificación, el análisis, la planificación de respuesta y el control de estos en el proyecto. Existen riesgos tanto positivos como negativos y dentro de cualquier proyecto debe intentar aumentar la probabilidad y el impacto de aquellos eventos positivo a la vez de disminuir la probabilidad y el impacto de los negativos. El riesgo dentro de un proyecto es un evento incierto que de producirse tiene un efecto bien positivo o negativo en alguno de los objetivos del proyecto (cronograma o tiempo, coste, alcance y/o calidad).

Los riesgos tienen origen en la incertidumbre del proyecto. Los riesgos conocidos, por el hecho de ser conocidos son aquellos que se pueden identificar y analizar posibilitando su planificación y respuesta. Los riesgos no conocidos, sin embargo, no pueden gesti0narse y de materializarse se considera un problema.

A continuación, se detallan esos riesgos conocidos aplicando las pautas para la dirección de proyectos que indica la Guía de PMBOK®. Aunque la gestión de los riesgos va más allá y contempla su planificación, su identificación, el análisis cuantitativo y cualitativo, su control y la planificación de las respuestas a los mismos si se producen; tan sólo va a realizarse una identificación, el análisis cualitativo haciendo uso de la herramienta online [http://www.itmplatform.com](http://www.itmplatform.com/) representando la matriz de evaluación de riesgos y la respuesta y la estrategia de respuesta al riesgo.

La estrategia de respuesta al riesgo será en función de si se observa como amenaza siendo las posibles estrategias:

* Evitar: eliminar la amenaza eliminando la causa.
* Mitigar: reducir impacto o probabilidad.
* Transferir: desviar o que terceros asuman la responsabilidad de estos riesgos

o por el contrario las opciones de estrategia de respuesta cuando el riesgo se trata de una oportunidad:

* Explotar: hacer que la oportunidad ocurra (contrario a evitar)
* Mejorar: Aumentar el impacto o probabilidad (contrario a mitigar)
* Compartir: Hacer dueño a terceros la posibilidad de lograr la oportunidad.

A continuación se muestran los valores definidos para el impacto y la probabilidad y los umbrales de nivel que posteriormente serán mostrados en las matrices de los riesgos identificados.

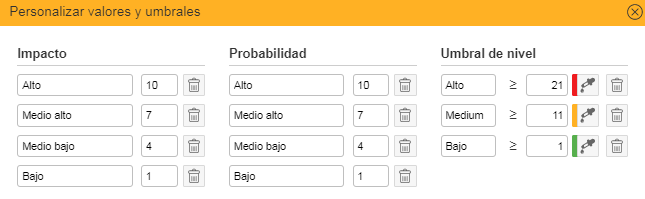


Ilustración 14: Valores Impacto y Probabilidad

## Riesgos del proyecto.

Son los riesgos asociados a la estimación, la planificación, al control del proyecto o a la comunicación.

|  |
| --- |
|  |
| Identificación del riesgo: Cambio de alcance.  Impacto: Retraso en la entrega. Entregables no adecuados.  Responsable: Gestor del proyecto  Respuesta: En caso de que se materialice debe comunicarse para realizar un control integrado de cambios donde se pueda replanificar el proyecto (tiempos, alcance, requisitos)  Estrategia de respuesta: Evitar/Mitigar |
| Identificación del riesgo: Estimación incorrecta de las tareas.  Impacto: Retraso en la entrega.  Responsable: Desarrollador/Gestor de proyecto  Respuesta: En caso de que se materialice debe comunicarse al gestor de proyecto o al cliente para replanificar nuevas fechas.  Estrategia de respuesta: Mitigar |
| Identificación del riesgo: Comprensión inadecuada o superficial de los requisitos.  Impacto: Entregables defectuosos por no cumplir los requisitos solicitados. Retraso en la entrega. Pérdida de confianza entre cliente y proveedor.  Responsable: Desarrollador/Gestor de Proyecto  Respuesta: En caso de que se materialice debe comunicarse para agendar sesiones de trabajo para aclarar conceptos, necesidades o requisitos.  Estrategia de respuesta: Evitar/mitigar. |
| Identificación del riesgo: Escasa o falta de comunicación entre los involucrados en el proyecto.  Impacto: Retrasos en la entrega, desmotivación en el proyecto, despriorización del proyetoco,  Responsable: Gestor de proyecto  Respuesta: En caso de que se materialice hacer un calendario de reuniones y sesiones de trabajo.  Estrategia de respuesta: Evitar |
| Identificación del riesgo: Despriorización del proyecto.  Impacto: Al despriorizar el proyecto y no tener interés puede provocar su no finalización y que los trabajos queden a medias.  Responsable: Cliente.  Respuesta: En caso de que se materialice comunicar para desasignar recursos y replanificar o cerrar definitivamente.  Estrategia de respuesta: Evitar |

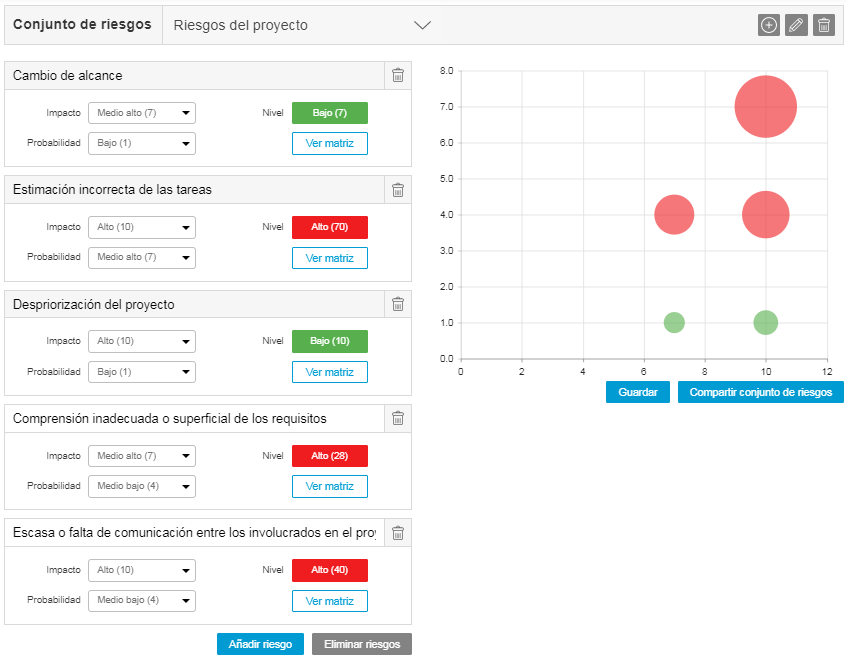


Ilustración 15: Matriz de evaluación de riesgos de proyecto

## Riesgos técnicos.

Son los riesgos que se relacionan con los requisitos a satisfacer, la tecnología que se aplique, la complejidad, al rendimiento y/o calidad de los entregables.

|  |
| --- |
|  |
| Identificación del riesgo: Falta de información pública y accesible de la situación actual de la tecnología sobre la que se desea hacer uso.  Impacto: Retraso en la entrega del trabajo.  Responsable: Gestor del proyecto.  Respuesta: En caso de que se materialice ampliar las búsquedas y fuentes de información.  Estrategia de respuesta: Evitar/Mitigar |
| Identificación del riesgo: Recursos con bajo conocimiento sobre la tecnología sobre la que se realiza el trabajo.  Impacto: Retraso en los desarrollos y en consecuencia en la entrega del trabajo o en su calidad.  Responsable: Desarrollador.  Respuesta: Ampliar el tiempo para autoformarse o buscar conocimiento en compañeros y cliente para asesoramiento.  Estrategia de respuesta: Mitigar |
| Identificación del riesgo: Compleja instalación y/o integración entre los productos seleccionados.  Impacto: Si la instalación se vuelve compleja por los productos en sí o la configuración de estos y su integración entre otros que conforman la solución final, puede conllevar retrasos en la entrega final.  Responsable: Desarrollador/Gestor de Proyecto  Respuesta: En caso de que se materialice debe comunicarse para modificar la solución final seleccionada.  Estrategia de respuesta: Evitar/mitigar. |
| Identificación del riesgo: Prueba de software insuficiente.  Impacto: Si no se testea lo suficiente el software puede generar problemas al cliente porque no satisfaga las necesidades.  Responsable: Desarrollador/Gestor de proyecto  Respuesta: En caso de que se materialice comunicar del riesgo y alargar el periodo de pruebas, replanificando las fechas del proyecto.  Estrategia de respuesta: Evitar |
| Identificación del riesgo: Desmotivación.  Impacto: Desmotivación acompañada de pérdida de interés hacia el trabajo, la implementación de la solución. La causa puede ser el poco avance del proyecto debido a su complejidad, que sean tareas poco motivantes, que se haya perdido el objetivo final…  Responsable: Desarrollador/Gestor de proyecto.  Respuesta: En caso de que se materialice dividir las tareas en hitos más cortos, mantener más sesiones con el cliente y para obtener feedbak positivo sobre el avance.  Estrategia de respuesta: Evitar |

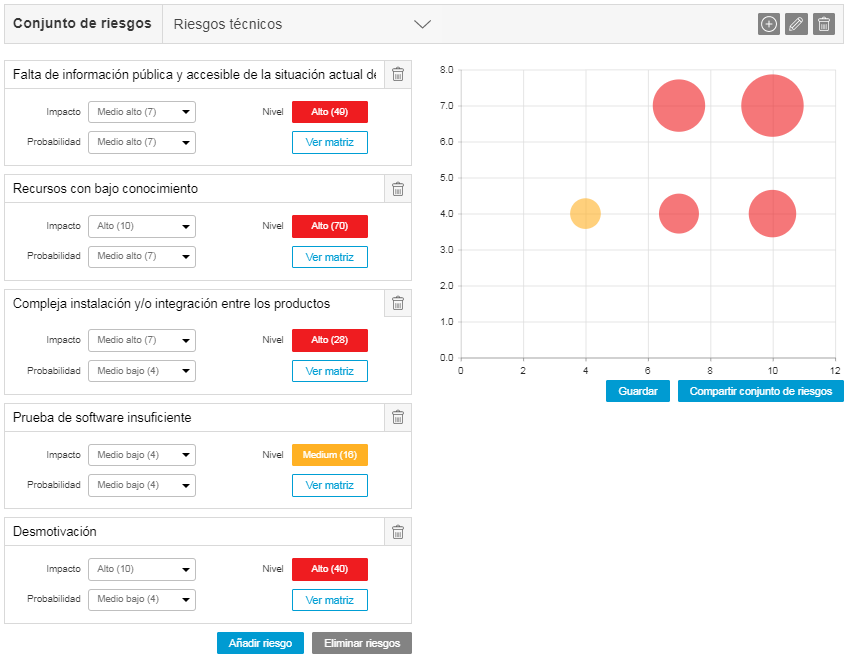


Ilustración 16: Matriz de evaluación de riesgos técnicos

## Riesgos externos.

Son aquellos riesgos que están relacionados con los proveedores, la normativa legal, el mercado, el cliente o el clima.

|  |
| --- |
| Identificación del riesgo: Versiones de los productos o herramientas utilizadas sean inestables y causen problemas.  Impacto: Si la versión de las herramientas de los proveedores no es estable y hay que esperar a que envíen un parche o saquen al mercado una versión estable puede ocasionar retrasos en la planificación y/o entregables de mala calidad o defectuosos.  Responsable: Externos.  Respuesta: En caso de que se materialice analizar otras alternativas como volver a versiones anteriores estables o cambiar de producto.  Estrategia de respuesta: Mitigar |
| Identificación del riesgo: Salida del patrocinador del proyecto.  Impacto: Si la persona que patrocina el proyecto sale del mismo debido a cambios en la organización o cualquier otro, el proyecto puede quedarse sin apoyo lo que puede provocar que no se finalice.  Responsable: Externos.  Respuesta: En caso de que se materialice buscar otro nuevo patrocinador, si no se encuentra, realizar la entrega en el estado actual y cerrar formalmente el proyecto.  Estrategia de respuesta: Mitigar |

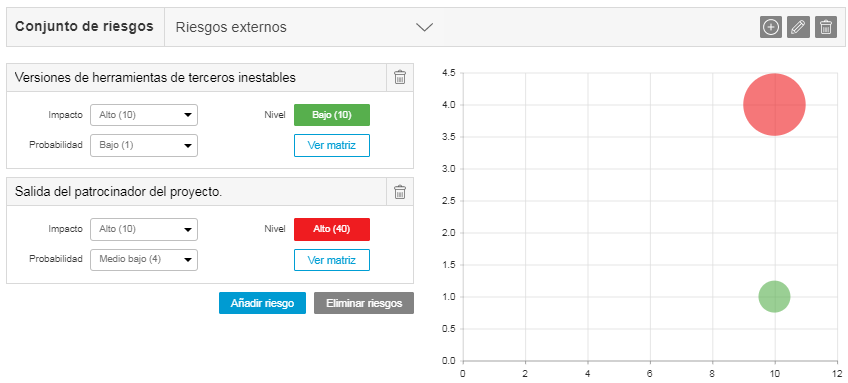
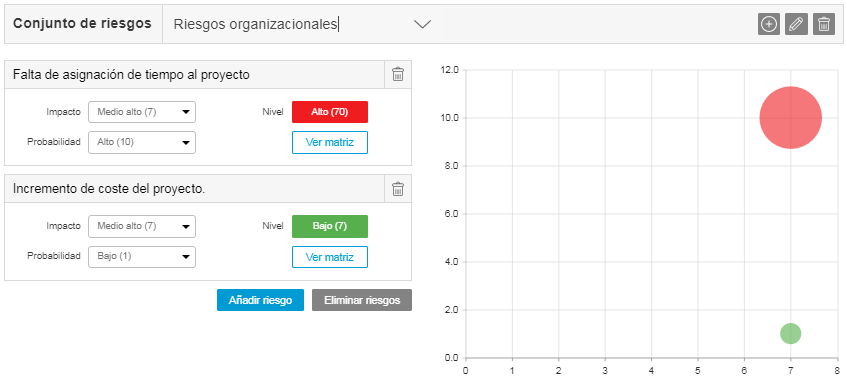


Ilustración 17: Matriz de evaluación de riesgos externos

## Riesgos organizacionales.

Riesgos derivados de las dependencias de los proyectos (interproyectos), los recursos, la priorización y/o la financiación del proyecto (costes).

|  |
| --- |
| Identificación del riesgo: Falta de asignación de tiempo del recurso al proyecto.  Impacto: Cuando el recurso asignado no está al 100% de su tiempo y se dedica a otras tareas o proyecto puede generar una sobre demanda que exceda a sus capacidades causando retrasos en el proyecto.  Responsable: Terceros (la organización)  Respuesta: En caso de que se materialice intentar que ceda su patrocinio a otro compañero de su departamento.  Estrategia de respuesta: Mitigar |
| Identificación del riesgo: Incremento de coste del proyecto.  Impacto: Si los proveedores pasan de licencias gratuitas a ser de pago en sus productos, el incremento del coste del proyecto será mayor.  Responsable: Terceros (el/los proveedor/es)  Respuesta: En caso de que se materialice analizar con el cliente las vías alternativas.  Estrategia de respuesta: Mitigar |



Como resumen podemos concluir que en base a la matriz de evaluación de los riesgos tenemos el siguiente listado ordenado de los riesgos según su nivel de exposición en el proyecto:

* Estimación incorrecta de las tareas (70)
* Recursos con bajo conocimiento sobre la tecnología (70)
* Falta de asignación de tiempo del recurso al proyecto (70)
* Falta de información pública y accesible de la situación actual (49)
* Escasa o falta de comunicación entre los involucrados (40)
* Desmotivación (40)
* Salida del patrocinador del proyecto (40)
* Comprensión inadecuada o superficial de los requisitos (28)
* Compleja instalación y/o integración entre los productos (28)
* Prueba de software insuficiente (16)
* Despriorización del proyecto (10)
* Versiones de los productos o herramientas utilizadas sean inestables (10)
* Cambio de alcance (7)
* Incremento de coste del proyecto (7)

1. Motor de búsqueda Elasticsearch

### Lucene Core.

Antes de analizar Elasticsearch como motor de búsqueda se presenta en qué consiste Lucene ya que esta solución se sustenta sobre él como muchos otros motores de búsqueda.



Lucene Core forma parte del proyecto Apache Lucene TM, un proyecto de Apache Fundation que nace en 1999 aplicando los conocimientos sobre NoSQL y desarrollando un software de código abierto disponible para descarga gratuita. Aunque en muchas ocasiones se confunde con un motor de búsqueda puesto que permite búsquedas e indexación de documentos, no lo es.

El subproyecto Lucen Core, es una biblioteca de códigos y una API para la recuperación de información de búsqueda ultrarrápida y no una aplicación completa. Esta librería ofrece tecnología de búsqueda e indexación, corrección ortográfica, resaltado de resultados y capacidades avanzadas de análisis, con lo que puede usarse fácilmente para agregar capacidades de búsqueda a las aplicaciones que la contengan. Supone así, un estándar de alto rendimiento para las búsqueda e indexación a la vez que una solución escalable. Aunque originalmente está basada en java también cuenta con versiones para otros lenguajes como Perl, C#, C++, Python, PHP, .NET entre otros.

Dentro de las características de Lucene se destacan:

* Indexación escalable de alto rendimiento.
* Algoritmos de búsqueda potentes, precisos y eficientes.
* Solución multiplataforma.

Lucene permite indexar, es decir, hacer índices; ordenar una serie de datos o informaciones de acuerdo a un criterio común a todos ellos, para facilitar su consulta y análisis y también realizar búsquedas sobre cualquier conjunto de datos de los que se pueda extraer el texto todo ello, abstrayéndose de la fuente, el formato del mismo y el idioma. Esto permite que puedan incluirse en este proceso, desde documentos de Word o archivos PDF hasta contenido de páginas web en formato HTML o cualquier otro formato del que pueda sacarse información textual.

Los procesos que conforman el núcleo del funcionamiento de Lucene pueden dividirse en:

***Proceso de análisis y tokenización***

El proceso de análisis consiste en convertir los campos de texto del contenido suministrado ya bien sean documentos pdf, documentos word, contenido web, etc en unidades para su representación en el índice, estas unidades de representación son llamadas términos. Los índices se utilizarán más adelante para permitir las búsquedas y mostrar el contenido coincidente con el criterio de búsqueda introducido por el usuario. Los resultados obtenidos de las búsquedas serán los coincidentes que contienen información relevante para el usuario. En Lucene existen diferentes tipos de analizadores que permiten controlar este proceso. Los analizadores, son clases java que pueden personalizarse en función de los requisitos que deban cumplirse, crearse nuevos analizadores o extenderlos y ampliarlos haciendo uso de sus funcionalidades. El texto se puede dividir en cadenas con significado y posteriormente filtrarlos por ellos, de esta forma el proceso de creación de estas cadenas puede ser personalizado.

Las aplicaciones que desarrollan sus capacidades de búsqueda en Lucene pueden admitir documentos en varios formatos (HTML, XML, PDF, Word, etc..) y es responsabilidad de la aplicación que hace uso de la librería Lucene usar un analizador adecuado para convertir el formato original en texto sin formato antes de hacérselo llegar a Lucene.

El texto sin formato suministrado a Lucene para la indexación pasa por un proceso denominado tokenización. La tokenización es el proceso de dividir el texto de entrada en pequeños elementos de indexación: tokens. La forma en que el texto de entrada se divide en tokens influye en gran medida en cómo las personas podrán buscar ese texto. Por ejemplo, los comienzos y finales de las oraciones pueden identificarse para proporcionar búsquedas más precisas de frases y proximidad.

En algunos casos, simplemente dividir el texto de entrada en tokens no es suficiente, puede ser necesario un análisis más profundo. Lucene incluye instalaciones de análisis de tokenización anteriores y posteriores.

El análisis previo a la tokenización puede incluir entre otras cosas a eliminar el marcado o tags de HTML y transformar o eliminar texto que coincida con patrones o conjuntos de cadenas fijas.

Hay muchos pasos posteriores a la tokenización que se pueden realizar como son:

* Steamings. Mencionado anteriormente. Supondría un reemplazo de palabras por su raíz. Por ejemplo, “Azules” sería reemplazado por “azul” con lo que al consultar “azul” encontraría documentos que contuviesen tanto “azul” como “azules” o “azulado”. En idiomas como el castellano al no mantener siempre la raíz léxica en ocasiones no puede realizarse este paso.
* Stopwords: las palabras comunes o de uso frecuente como puedan ser “el”, “y” o “a” rara vez agregan valor a la búsqueda con lo que pueden ser eliminadas y la consecuencia a esta eliminación es la reducción del tamaño del índice, aumentan el rendimiento. También pueden reducirse ciertos “ruidos” del texto y mejorar la calidad de la búsqueda.
* [Normalización del texto](http://en.wikipedia.org/wiki/Text_normalization) : eliminación de los acentos y otros símbolos de caracteres mejorando la búsqueda.
* [Expansión de sinónimos](http://en.wikipedia.org/wiki/Synonym) : agregar sinónimos en la misma posición del token que la palabra actual puede significar una mejor concordancia cuando los usuarios buscan con palabras en el conjunto de sinónimos.

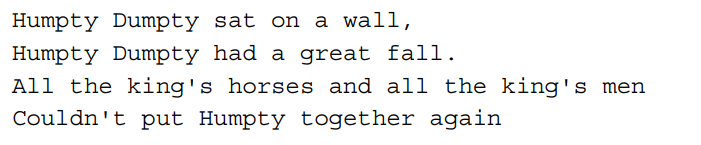
El paquete **org.apache.lucene.analysis** ofrece la librería para el análisis de texto que convierte el contenido o documentos que le llegan a tokens o cadenas con significado y filtros. Aquí, se encuentra el analizador estándar que está basado en gramática de uso general que implementa las reglas de separación de palabras del algoritmo de segmentación de texto Unicode. Es importante seleccionar el analizador adecuado ya que los resultados obtenidos dependen se debe tener en cuenta para su elección el idioma o las abreviaturas.

El analizador incluye operaciones como la extracción de los tokens, eliminación de los signos de puntuación, diéresis, acentos, normalización (conversión a minúsculas/mayúsculas) o stemming. También analizadores para distintos idiomas entre ellos un paquete propio para español **org.apache.lucene.analysis.es** con la clase SpanishAnalyzer o crear un propio Analizador según las necesidades de la aplicación:

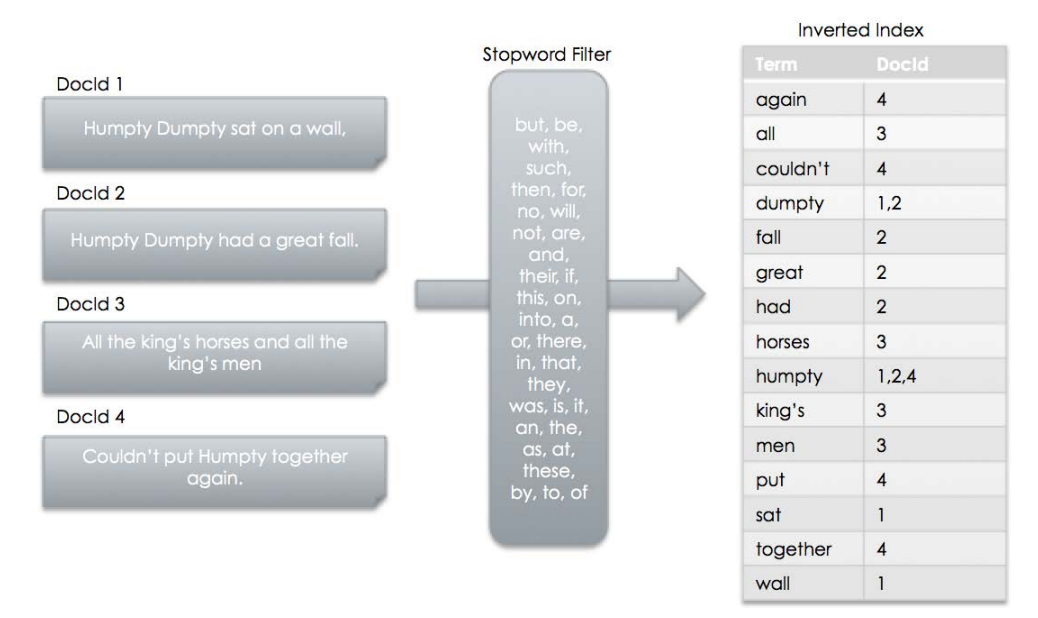
|  |  |
| --- | --- |
| Analyzer usando el builder | Analyzer usando nombres SPI |
| Analyzer ana = CustomAnalyzer.builder(Paths.get("/path/to/config/dir"))  .withTokenizer(StandardTokenizerFactory.class)  .addTokenFilter(StandardFilterFactory.class)  .addTokenFilter(LowerCaseFilterFactory.class)  .addTokenFilter(StopFilterFactory.class, "ignoreCase", "false", "words", "stopwords.txt", "format", "wordset")  .build(); | Analyzer ana = CustomAnalyzer.builder(Paths.get("/path/to/config/dir"))  .withTokenizer("standard")  .addTokenFilter("standard")  .addTokenFilter("lowercase")  .addTokenFilter("stop", "ignoreCase", "false", "words", "stopwords.txt", "format", "wordset")  .build(); |

**org.apache.lucene.document** es el paquete ofrecido por Lucene para la representación lógica de un Document para indexary buscar siendo éste un conjunto de nombres (Fields), el paquete **org.apache.lucene.codecs** proporciona una abstracción sobre la codificación/decodificación de la estructura del índice.

En el siguiente ejemplo, suponiendo que cada una de las líneas sea el contenido de cada uno de nuestros documentos:



Edwood Ng, Vineeth Mohan - Lucene 4 Cookbook\_ Over 70 hands-on recipes to quickly and effectiv



Este proceso de análisis se realiza en la fase de indexación y en la de búsqueda de documentos.

*Proceso de indexación de documentos.*

Es el núcleo de todos los motores de búsqueda. Consiste en añadir los documentos originales al índice convirtiéndolos en texto plano. Lucene almacena los documentos en una estructura de índice invertido optimizando el almacenamiento en disco y la velocidad de las búsquedas. Lucene divide los documentos en palabras o tokens que utiliza en el índice como claves de búsqueda. La indexación sería similar por ejemplo a los índices alfabéticos que tenemos al final de un libro donde se muestran palabras claves, relevantes o comunes con los números de página donde aparecen para que estas palabras puedan rastrearse rápidamente, en lugar de buscar en el libro completo.

Lucene funciona con frecuencia de término y frecuencia inversa de documento. Es decir, Mide con qué frecuencia aparece un término o palabra dentro de un documento determinado, y lo compara con el número de documentos que la mencionan dentro del conjunto total de documentos que posee.

Por ejemplo, palabras como “el” o “la” aunque aparecen con alta frecuencia dicen poco, puesto que al analizar el resto de documentos, estas palabras son muy comunes. Sin embargo, si un documento contiene repetidamente el término “mariposa”, las posibilidades de que ese documento sea relevante para la búsqueda “mariposa” crecen, ya que es poco frecuente dentro del conjunto total de documentos.

<incorporo fórmula de calculo de tf-i>

El paquete **org.apache.lucene.index** contiene dos clases principales para gestionar el indexado de documentos, estas son, **IndexWriter** que permite crear nuevos índices o modificar los que ya están existentes añadiendo, borrando o actualizando documentos (en este caso, primero elimina y luego añade el documento completo), es decir, crea y mantiene un índice, e **IndexReader** que accede a los datos en el índice y forma parte del proceso de búsqueda.

Con el **OpenMode** de la clase [**IndexWriterConfig** se](http://lucene.apache.org/core/8_0_0/core/org/apache/lucene/index/IndexWriterConfig.html#setOpenMode-org.apache.lucene.index.IndexWriterConfig.OpenMode-) determina si se crea un nuevo índice o si se abre un índice existente. Además pueden configurarse entre otros, el tamaño del buffer **DEFAULT\_RAM\_BUFFER\_SIZE\_MB** con un tamaño por defecto de 16MB determina la cantidad de RAM que se puede usar para almacenar los documentos añadidos y eliminados en el búfer antes de que se escriban en el Directorio.

……

Path indexPath = Files.createTempDirectory("tempIndex");

Directory directory = FSDirectory.open(indexPath)

IndexWriterConfig config = new IndexWriterConfig(analyzer);

IndexWriter iwriter = new IndexWriter(directory, config);

Document doc = new Document();

String text = "This is the text to be indexed.";

doc.add(new Field("fieldname", text, TextField.TYPE\_STORED));

iwriter.addDocument(doc);

iwriter.close();

………

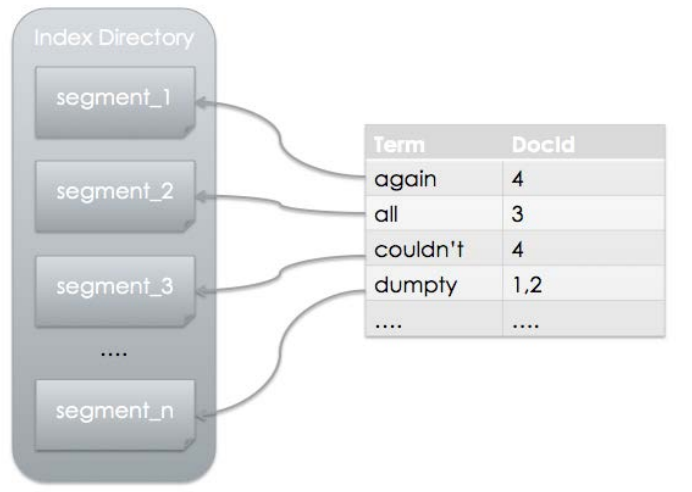
En este sencillo ejemplo de uso de la clase **IndexWriter**, observamos como es necesario tener definido un directorio para el índice, un analizador para realizar el análisis el contenido o documentos a indexar, el escritor y la configuración del miso y el documento a indexar.

Como se comentó anteriormente, en la fase de análisis, la clase Document también interviene en esta fase de indexación.

**IndexWriter** puede alterar los segmentos en un índice con lo que puede ser necesario forzar la fusión de todos los segmentos que tienen documentos eliminados o hasta que haya un máximo de números de segmentos.

En Lucene, del formato de archivo es de naturaleza segmentada. Los archivos de índice están separados físicamente por segmentos y siguiendo una nomenclatura estándar, segment\_1, segment\_2, etc. Todos los archivos que pertenecen al mismo segmento comparten el mismo nombre de archivo con diferentes extensiones. Cada segmento es un índice totalmente independiente que se puede buscar por separado. El mero hecho de añadir nuevos documentos puede generar un nuevo.

Los documentos pueden existir en cualquiera de los segmentos, dependiendo de cuándo sean añadidos los nuevos documentos. Los atributos de IndexWriter fusionarán los segmentos del índice internamente para que pueda acceder a un índice.



Lunece requiere de la creación de objetos de tipo Document. Los documentos son la unidad de indexación y búsqueda. Un documento de Lucene contiene o está formado por uno o más campos que lo identifican de manera única. Un campo podría ser el título, la descripción, el identificador del documento, etc. Un campo de Lucene tiene tres atributos:

* Nombre
* Valor
* Tipo

El tipo de campo determina como será tratado el mismo, se puede definir el tipo (FieldType) para controlar si almacenar el valor, para indexarlo o tokenizaar el texto. Lucene admite Strings, Readers o TokenStream, binarios o valores numéricos.

##### Proceso de búsqueda

La búsqueda es un proceso que consiste en encontrar términos en un índice para acceder a los documentos. para esta fase de es necesario algunos prerequisitos, como tener el directorio de los índices abierto, configurado el analizador y haber escrito en los índices para preparar la búsqueda.

**IndexReader** forma parte del paquete **org.apache.lucene.index** que ya vimos antes y nos permite acceder a una vista de un punto en el tiempo de un índice (point-in-time). Con este tipo de acceso se puede escribir simultáneamente en un índice, mientras que un **IndexReader** está leyendo sin exponer los datos no confirmados.  Cualquier cambio realizado en el índice vía **[IndexWriter](http://lucene.apache.org/core/8_0_0/core/org/apache/lucene/index/IndexWriter.html" \o "clase en org.apache.lucene.index)** no será visible hasta que **IndexReader** abra un nuevo índice. Cuando se necesita ver los cambios realizados en el índice, es mejor hacerlo a través de un método específico de **DirectoryReader**. **DirectoryReader** es una subclase de **IndexReader**, que es la clase que proporciona la facilidad para abrir un directorio que además dispone de un método de apertura de índice más optimizado llamado **openIfChanged** que reutilizará el **DirectoryReader** existente para una reapertura más rápida.

Existen dos tipos diferentes de **IndexReaders**:

* **LeafReader**: Se trata de índices atómicos. Admiten la recuperación de campos almacenados, valores de documentos, términos y publicaciones.
* **CompositeReader**: Las instancias de este lector (como **DirectoryReader**) solo pueden usarse para obtener campos almacenados de los **LeafReaders**.

Por eficiencia, se hace referencia a los documentos mediante números naturales que relacionan cada documento único en el índice. Estos números de documentos son efímeros; pueden cambiar a medida que los documentos se agregan y se eliminan de un índice. Por lo tanto, no se debe confiar en que un documento dado tenga el mismo número entre sesiones.

Una búsqueda consiste esencialmente en encontrar la intersección del identificador del documento (DocId) con los términos coincidentes. El índice en sí mismo permite una búsqueda de términos muy rápida, pero no tanto para la búsqueda por DocId. Para resolver el problema de encontrar un valor de campo por DocId, Lucene introdujo FieldCache. FieldCache es una estructura de datos en memoria que se almacena en un formato de matriz en el que la posición del valor corresponde a DocId (ya que DocId es básicamente un valor ordinal de todos los documentos). Debido a que cada posición solo puede almacenar un valor, FieldCache debe usarse solo en campos de valor único. En esencia, Lucene desinvierte los datos del índice y los almacena en FieldCache. El FieldCache es estático y no está sincronizado con el IndexReader con lo que deberá ser reinicializado para estar sincronizado con los últimos cambios.

Antes de comenzar la búsqueda es necesario obtener un **IndexSearcher** que nos facilitará las consultas en un índice determinado. Esta clase se encuentra dentro del paquete org.apache.lucene.search, este paquete además, contiene las estructuras de datos para representar consultas.

* TermQuery: para palabras y términos individuales
* PhraseQuery: para frases.
* BooleanQuery: para las combinaciones booleanas de las consultas.
* QueryParser: exisen varios tipos para producir estructuras de consulta a partir de cadenas o xml.

**IndexSearcher** contiene un conjunto de métodos para consultas de datos y el resultado lo de vuelve en clases **TopDocs**. Éstos representan los ‘hits’ o coincidencias en forma de arrays de **ScoreDocs** que contiene el DocId y la puntuación de cada documento. Estos **TopDocs** como se ha dicho contiene el identificador del documento y no su contenido, para recuperarlo seberá realizarse mediante la clase **IndexReader** o **FieldCache**.

Para construir la consulta de búsqueda podemos hacerlo de distintos modos, programando el objeto **Query** o a través de **QueryParser** que proporciona la utilidad de interpretar el texto y convertirlo en **Query**.

Ejemplo.

Otras funcionalidades avanzadas de la búsqueda:

* Ordenación, puede ordenarse por puntuación de relevancia, por orden en el índice (ordenamiento natural del documento en el índice) o personalizar esta ordenación creando propia lógica de comparación.
* Paginación, tradicionalmente se muestran los resultados de una consulta guardándolos en memoria e iterándolos cuando se visualizan, de esta forma se minimizan las consultas del motor de búsqueda, sin embargo, en Lucene.
* Búsqueda en Near Real Time (NRT). Lucene posee la capacidad de dejar disponibles para búsqueda y su consulta los documentos inmediatamente después de haber sido indexados.
* Filtros. Aunque os filtros parecen un término similar al de consultas, la diferencia radica en que las consultas tienen en cuenta las puntuaciones calculadas y la relevancia de las palabras o términos a la hora de realizar las búsquedas mientras que los filtros no tienen efecto en las puntuaciones. Los usuarios en ocasiones prefieren navegar entre filtros para llegar a su contenido relevante. Lucene provee de un conjunto de filtros diseñados para ser usados por las aplicaciones y además permite construir los propios extendiendo la clase **Filter**, si es que los que ofrece no se ajustan, entre otros:
  + **TermRangeFilter**: filtro para restringir los resultados a un conjunto de términos definidos entre un límite inferior y superior al rango enviado.
  + **NumericRangerFilter**: restringe los resultados a un rango o número de valores.
  + **FildCacheRangerFilter**: se ejecuta sobre el rango de filtros y almacena en memoria los resultados para mejorar el rendimiento.
  + **FieldValueFilter**: devuelve un documento que contiene uno o más valores para un campo.
  + **PrefixFilter**: limita los resultados que coinciden con el prefijo definido.
* Puntuación. Normalmente no vemos la puntuación porque los documentos son ordenados por relevancia.

Una vez revisado la forma en la que trabaja la librería Lucene es el momento de ver los dos motores de búsqueda que se va a evaluar y analizar como alternativas a tener en cuenta para la instalación final de la solución, estos son Elasticsearch y Solr. Se abordará en primer lugar por la herramienta Solr por cercanía a la librería expuesta y por pertenener a parte del mismo proyecto.

A modo resumen, para que una aplicación pueda usar Lucene como librería, ésta debe:

* Crear Documents añadiendo Fields.
* Crear IndexWriter y agregar los documentos a él con addDocument()
* Llamar a QueryParser.parse() para generar una consulta a partir de una cadena y
* Crear IndexSearcher y pasar la consulta al método search()

Los motores de búsqueda como se vió anteriormente, hacen referencia a una gran base de datos de recursos como páginas web, grupos de noticias, programas, imágenes, documentos etc. Y que ayudan a localizar esta información almacenada.

Los usuarios pueden buscar información mediante consultas realizadas al motor de búsqueda en forma de palabras clave o frases, el motor de búsqueda, realiza la consulta suministrada por el usuario y busca en el conjunto de datos almacenados devolviendo aquella información relevante al usuario.

<https://db-engines.com/en/ranking/search+engine>

Tanto Elasticsearch como solr, según DB-Engines son 2 de los tres motores de búsqueda más mencionados y populares en internet.

A grandes rasgos los motores de búsqueda funcionan siguiendo los pasos:

**Obtener contenido en bruto.** Este es el primer paso de una aplicación de búsqueda, debe recopilar los contenidos sobre los que se realizarán las búsquedas.

**Construir el documento.** Construir el documento a partir del contenido en bruto para que la aplicación de búsqueda pueda entenderlo e interpretarlo fácilmente.

**Analizar el documento.** Seguidamente, se debe pasar por un proceso de análisis sobre el que se transforma el documento antes de ser indexado.

**Indexar el documento.** Una vez creados y documentos y analizados se procede a su indexado para que su recuperación pueda realizarse en función de ciertas claves.

**Interfaz de usuario para la búsqueda. Para** ayudar al usuario a realizar la búsqueda, la aplicación debe proporcionar una interfaz de usuario donde el usuario pueda introducir el texto a consultar e iniciar el proceso de búsqueda.

**Construir la consulta.** Cuando el usuario realiza la solicitud para buscar un texto, la aplicación debe preparar esta consulta para localizar los documentos coincidentes em el índice y así obtener los resultados relevantes.

**Buscar la consulta.** Con la consulta preparada, se realiza la búsqueda para obtener los contenidos relevantes y se obtienen los documentos.

**Mostrar los resultados.** Una vez obtenido el resultado, la aplicación debe mostrar los resultados al usuario a través de la interfaz de usuario.

Mientras que Lucene no implementa los pasos primero y último, los motores de búsqueda suponen una capa superior que incluye la totalidad de ellos.

Conceptos de Elastichsearch

**Clúster**. Un clúster es una colección de uno o más nodos (servidores) que, en conjunto, conservan sus datos completos y proporcionan capacidades de búsqueda y indexación federada en todos los nodos. Un clúster se identifica con un nombre único que, de forma predeterminada, es "elasticsearch". Este nombre es importante porque un nodo solo puede ser parte de un clúster si el nodo está configurado para unirse al clúster por su nombre.

Asegúrese de no reutilizar los mismos nombres de clúster en diferentes entornos, de lo contrario podría terminar con nodos que se unan al clúster incorrecto. Por ejemplo, podría utilizar logging-dev, logging-stagey logging-prod para los grupos de desarrollo, organización y producción.

Tenga en cuenta que es válido y perfectamente correcto tener un clúster con un solo nodo en él. Además, también puede tener varios clústeres independientes, cada uno con su propio nombre de clúster único.

Nodo. Un nodo es un servidor único que forma parte de su clúster, almacena sus datos y participa en las capacidades de búsqueda e indexación del clúster. Al igual que un clúster, un nodo se identifica con un nombre que, de forma predeterminada, es un identificador universal único aleatorio (UUID) que se asigna al nodo en el inicio. Puede definir cualquier nombre de nodo que desee si no desea el valor predeterminado. Este nombre es importante para fines de administración en los que desea identificar qué servidores de su red se corresponden con los nodos de su grupo de Elasticsearch.

Un nodo se puede configurar para unirse a un clúster específico por el nombre del clúster. De forma predeterminada, cada nodo está configurado para unirse a un clúster denominado, lo elasticsearchque significa que si inicia una serie de nodos en su red y, suponiendo que puedan descubrirse unos a otros, todos se formarán automáticamente y se unirán a un único clúster llamado elasticsearch.

En un solo clúster, puede tener tantos nodos como desee. Además, si no hay otros nodos de Elasticsearch actualmente en ejecución en su red, el inicio de un solo nodo de forma predeterminada formará un nuevo clúster de un solo nodo llamado elasticsearch.

Índice. Un índice es una colección de documentos que tienen características similares. Por ejemplo, puede tener un índice para datos de clientes, otro índice para un catálogo de productos y otro índice para datos de pedidos. Un índice se identifica con un nombre (que debe estar todo en minúsculas) y este nombre se usa para referirse al índice cuando se realizan operaciones de indexación, búsqueda, actualización y eliminación en los documentos que contiene.

En un solo clúster, puede definir tantos índices como desee.

Un documento es una unidad básica de información que puede ser indexada. Por ejemplo, puede tener un documento para un solo cliente, otro documento para un solo producto y otro para un solo pedido. Este documento se expresa en JSON (Notación de objetos de JavaScript), que es un formato de intercambio de datos de Internet ubicuo. Dentro de un índice, puede almacenar tantos documentos como desee.

### Fragmentos y réplicas[editar](https://github.com/elastic/elasticsearch/edit/7.0/docs/reference/getting-started.asciidoc)

Un índice puede almacenar potencialmente una gran cantidad de datos que pueden exceder los límites de hardware de un solo nodo. Por ejemplo, es posible que un solo índice de mil millones de documentos que ocupan 1 TB de espacio en disco no se ajuste en el disco de un solo nodo o que sea demasiado lento para atender las solicitudes de búsqueda desde un solo nodo.

Para resolver este problema, Elasticsearch brinda la capacidad de subdividir su índice en múltiples partes llamadas fragmentos. Cuando crea un índice, simplemente puede definir la cantidad de fragmentos que desea. Cada fragmento es en sí mismo un "índice" completamente funcional e independiente que se puede alojar en cualquier nodo del clúster.

Sharding es importante por dos razones principales:

* Te permite dividir / escalar horizontalmente tu volumen de contenido
* Le permite distribuir y paralelizar operaciones entre fragmentos (potencialmente en múltiples nodos), lo que aumenta el rendimiento / rendimiento

Elasticsearch gestiona completamente la mecánica de cómo se distribuye un fragmento y también la forma en que sus documentos se agregan a las solicitudes de búsqueda y son transparentes para usted como usuario.

En un entorno de red / nube donde se pueden esperar fallas en cualquier momento, es muy útil y altamente recomendable tener un mecanismo de conmutación por error en caso de que un fragmento / nodo se desconecte o desaparezca por alguna razón. Para este fin, Elasticsearch le permite hacer una o más copias de los fragmentos de su índice en lo que se conoce como fragmentos de réplica, o réplicas para abreviar.

La replicación es importante por dos razones principales:

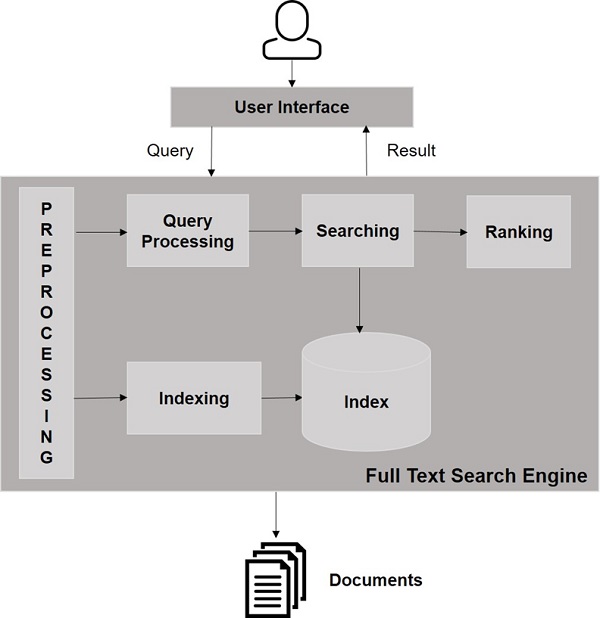
* Proporciona alta disponibilidad en caso de que un fragmento / nodo falle. Por este motivo, es importante tener en cuenta que un fragmento de réplica nunca se asigna en el mismo nodo que el fragmento original / primario del que se copió.
* Le permite escalar su volumen de búsqueda / rendimiento ya que las búsquedas se pueden ejecutar en todas las réplicas en paralelo.

Para resumir, cada índice se puede dividir en varios fragmentos. Un índice también se puede replicar en cero (es decir, sin réplicas) o más veces. Una vez replicado, cada índice tendrá fragmentos primarios (los fragmentos originales desde los que se replicaron) y fragmentos de réplica (las copias de los fragmentos primarios).

El número de fragmentos y réplicas se puede definir por índice en el momento en que se crea el índice. Después de crear el índice, también puede cambiar el número de réplicas dinámicamente en cualquier momento. Puede cambiar el número de fragmentos para un índice existente usando las API [\_shrink](https://www.elastic.co/guide/en/elasticsearch/reference/7.0/indices-shrink-index.html)y [\_split](https://www.elastic.co/guide/en/elasticsearch/reference/7.0/indices-split-index.html), sin embargo, esto no es una tarea trivial y la planificación previa para el número correcto de fragmentos es el enfoque óptimo.

De forma predeterminada, a cada índice en Elasticsearch se le asigna un fragmento primario y una réplica, lo que significa que si tiene al menos dos nodos en su grupo, su índice tendrá un fragmento primario y otro fragmento de réplica (una réplica completa) para un total de dos Fragmentos por índice.

Cada fragmento de Elasticsearch es un índice de Lucene. Hay un número máximo de documentos que puede tener en un solo índice de Lucene. A partir de ese momento [LUCENE-5843](https://issues.apache.org/jira/browse/LUCENE-5843), el límite es 2,147,483,519(= Integer.MAX\_VALUE - 128) documentos. Puedes monitorear los tamaños de fragmentos usando la [\_cat/shards](https://www.elastic.co/guide/en/elasticsearch/reference/7.0/cat-shards.html)API.



# Diseño.

## Análisis del Sistema.

# Implementación.

# Resultados y discusión.

# Conclusiones.

# Bibliografía

[MEA, 1992] Meadow, C. T. *Text Information retrieval Systems*. San Diego: Academic Press, 1993.

[SAL, 1983] Salton , G. and Mc Gill, M.J. *Introduction to Modern Information Retrieval*. New York: Mc Graw-Hill Computer Series, 1983.

Abadal Falgueras, Ernest; Codina Bonilla, Lluís. *Bases de datos documentales: características, funciones y método*. Madrid: Síntesis, DL 2005. 220 p. (Biblioteconomía y Documentación) https%3a%2f%2fwww.lluiscodina.com%2fwp-content%2fuploads%2friv2.doc

UNIVERSIDAD DE GRANADA. Modelos de sistemas de recuperación de información Documental Basados en información lingüística difusa [en línea].<http://sci2s.ugr.es/publications/ficheros/tesis_difinitiva.pdf> [citado en septiembre 11 de 2011]

<http://glossarium.bitrum.unileon.es/Home/recuperacion-de-informacion>

<https://vdiazlindsay.blogspot.com/2017/02/tema-1-introduccion.html>

<https://medium.com/@anayaotto/bases-de-datos-nosql-aplicado-a-motores-de-b%C3%BAsqueda-a39048f9963a>

<https://db-engines.com/en/ranking/search+engine>

<http://lucene.apache.org/>

<https://builds.apache.org/view/All/job/Lucene-Artifacts-master/javadoc/>

<https://lucene.apache.org/core/8_0_0/core/index.html?overview-summary.html>

<https://lucene.apache.org/solr/guide/7_7/index.html>

<https://wiki.apache.org/solr/PublicServers>

<https://www.tutorialspoint.com/apache_solr/apache_solr_search_engine_basics.htm>

[https://www.elastic.co](https://www.elastic.co/)

<http://json.org/>

<https://dev.mysql.com/downloads/file/?id=485812>

https://www.draw.io

https://db-engines.com/en/ranking

file:///D:/Users/abuenogi/Downloads/29595-94201-1-PB.pdf

https://info.aiim.org/aiim-blog/newaiimo/2012/03/15/big-data-and-big-content-just-hype-or-a-real-opportunity

https://db-engines.com/en/ranking\_trend/system/Elasticsearch%3BSolr

https://db-engines.com/en/system/Elasticsearch%3BSolr

file:///D:/Users/abuenogi/Downloads/74-221-2-PB.pdf

https://comparisons.financesonline.com/apache-solr-vs-elasticsearch

https://sematext.com/blog/solr-vs-elasticsearch-differences/

http://solr-vs-elasticsearch.com/

https://trends.google.com/trends/explore?date=all&q=apache%20solr,elasticsearch

https://thishosting.rocks/comparing-elasticsearch-with-solr/

**Referencias**

* ANTONIOU, G., VAN HARMELEN, F. (2004). A semantic Web Primer. Massachussets: MIT, 2004
* BAEZA-YATES, R., RIBEIRO-NETO, B. (1999). *Modern information retrieval*. New York : ACM Press ; Madrid[etc.]:  Addison-Wesley.
* CLEVERDON, C.W. (1972). “On the inverse relationship of recall and precision”. *Journal of Documentation*, Vol. 28, pp. 195-201.
* SPARCK, J. (1997). Readings in information retrieval . edited by Karen Sparck Jones, Peter Willett.San Francisco : Morgan Kaufmann.

Presentación:

<https://blog.irontec.com/apache-solr-tutorial/>

<http://www.cmsmatrix.org/>

https://www.gartner.com/doc/reprints?id=1-67R0V6S&ct=190211&st=sb

https://www.gartner.com

<https://www.cignex.com/blog/liferay-dxp-vs-liferay-7-community-edition-what%E2%80%99s-difference>

Libro de Rita de PMP

Project Management Institute. A Guide to the Project Management Body of Knowledge (PMBOK Guide). Technical Report ANSI/PMI 99-001-2004, American National Standard, 2004.

<http://www.itmplatform.com/es/recursos/matriz-de-evaluacion-de-riesgos/>