**Sistemas de Información Orientados a Servicios**

Trabajo final: Twico

Máster Universitario en Ingeniería Informática

diciembre de 2020



**Autores**

Miguel Cabezas Puerto

Luis Blázquez Miñambres

Francisco Pinto Santos

Oscar Sánchez Juanes

En este documento se recoge la memoria desarrollada por los alumnos Miguel Cabezas Puerto, Luis Blázquez Miñambres , Francisco Pinto Santos y Óscar Sánchez Juanes del trabajo final de la asignatura “Sistemas de Información Orientados a Servicios” en el seno del Máster en Ingeniería Informática de la Universidad de Salamanca en el curso 2020-2021, consistente en una aplicación web para el tratamiento y gestión de temas relacionados con el virus SARS-COVID-19, presentando diversas tecnologías que ayuden a mostrar una perspectiva más general y simplificada de su implicación en el día a día de la sociedad..

Contenido

[Índice de tablas 2](#_Toc59014141)

[Índice de ilustraciones 3](#_Toc59014142)

[1. Introducción 4](#_Toc59014143)

[2. Objetivos del proyecto 1](#_Toc59014144)

[3. Modelado del sistema 1](#_Toc59014145)

[3.1. Arquitectura del sistema 1](#_Toc59014146)

[3.2. Disposición de ETLs 1](#_Toc59014147)

[3.3. APIs utilizadas 1](#_Toc59014148)

[4. Conceptos teóricos 1](#_Toc59014149)

[4.1. ETL (Extract, Transform and Load) 1](#_Toc59014150)

[4.2. Web Scraping 2](#_Toc59014151)

[4.3. NLP 2](#_Toc59014152)

[4.4. Algoritmo LDA 2](#_Toc59014153)

[5. Tecnologías y herramientas utilizadas 1](#_Toc59014154)

[5.1. Lenguajes de programación 1](#_Toc59014155)

[5.1.1. Python 1](#_Toc59014156)

[5.1.2. VueJS 2](#_Toc59014157)

[5.1.3. Node.js 2](#_Toc59014158)

[5.1.4. TypeScript 3](#_Toc59014159)

[5.2. VaderSentiment 3](#_Toc59014160)

[5.3. Scikit-learn 3](#_Toc59014161)

[5.4. PM2 3](#_Toc59014162)

[5.5. Amcharts 3](#_Toc59014163)

[6. Funcionamiento del sistema 1](#_Toc59014164)

[Bibliografía 1](#_Toc59014165)

# Índice de tablas

# Índice de ilustraciones

[Ilustración 1: Esquema ETL 1](#_Toc59014216)

[Ilustración 1: Logo de Python 1](#_Toc59014217)

[Ilustración 2: Logo de Node.js 2](#_Toc59014218)

# Introducción

# Objetivos del proyecto

Para el cumplimiento de las tareas y tecnologías planteadas dentro de la arquitectura del sistema, se marcaron una serie de objetivos a realizar para llegar a un producto que pudiera ser mantenido con el paso del tiempo y que fuera de utilidad, aplicando las diversas tecnologías que se explicarán posteriormente y con las cuales , los miembros del equipo del proyecto, trabajan y colaboran de forma asidua.

# Modelado del sistema

En este apartado se definirá, de forma más abstracta, el diseño y planteamiento de la arquitectura del sistema, explicando en detalle los componentes que lo conforman, diferentes variables a tener en cuenta para su disposición y despliegue, así como un listado con el conjunto de APIs públicas utilizadas con motivo de la asignatura y que han sido fuente de datos principal dentro de la lógica de negocio de la aplicación.

## Arquitectura del sistema

## Disposición de ETLs

## Componentes externos

### APIs

Para la gestión y funcionamiento del sistema se han hecho uso de diferentes APIs externas para la extracción de datos, todos acerca de la COVID-19. A continuación, se describirán cada una de las APIs utilizadas para el funcionamiento de la aplicación, el formato de los datos que devuelven y el objetivo de uso dentro del sistema.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **API** | **Endpoint** | **Descripción** | **Entrada** | **Salida** |
| Twitter API | https://api.twitter.com/1.1/search/tweets.json | Devuelve la información referente a los datos de los tweets acerca de un topic o *query* dada | * **Query**: palabra o conjunto de palabras por las que filtrar la búsqueda de tweets * **Lang**: idioma de los tweets a recoger * **from**: fecha de inicio de búsqueda de tweets * **to**: fecha de fin de búsqueda tweets | Debido a la gran extensión del formato de salida de los datos, si se desea consultar el formato de salida de los datos de este endpoint acceder al siguiente enlace <https://developer.twitter.com/en/docs/twitter-api/v1/tweets/search/api-reference/get-search-tweets> |
| NewsAPI | http://newsapi.org/v2/top-headlines | Devuelve la informació relativa las cabeceras de noticias asociadas a un país y a una categoría | * **Country**: país de origen de las noticias * **Category**: categoría de las noticias (deportes, salud, etc.) * **apiKey**: en formato texto la clave de la API para acceder a la información | Debido a la gran extensión del formato de salida de los datos, si se desea consultar el formato de salida de los datos de este endpoint acceder al siguiente enlace <https://developer.twitter.com/en/docs/twitter-api/v1/tweets/search/api-reference/get-search-tweets> |
| http://newsapi.org/v2/everything | Devuelve los datos asociados a las noticias que contengan una palabra o conjunto de palabras de una búsqueda dada. | * **query**: topic con la palabra o conjunto de palabras para filtrar las noticias * **from**: fecha de inicio de búsqueda de noticias * **to**: fecha de fin de búsqueda noticias * **sortBy**: punto de filtrado de las noticias (ascendente, descendente, …) * **apiKey**: en formato texto la clave de la API para acceder a la información | Debido a la gran extensión del formato de salida de los datos, si se desea consultar el formato de salida de los datos de este endpoint acceder al siguiente enlace <https://developer.twitter.com/en/docs/twitter-api/v1/tweets/search/api-reference/get-search-tweets> |
| OpenData | https://opendata-ajuntament.barcelona.cat/data/api/action/datastore\_search\_sql | Devuelve toda la información de datos públicos acerca de la COVID-19 en la ciudad de Barcelona en formato encolumnado | * **sql:** cadena de texto que contiene la consulta en formato SQL sobre la búsqueda de datos públicos que se va a realizar | Debido a la gran extensión del formato de salida de los datos, si se desea consultar el formato de salida de los datos de este endpoint acceder al siguiente enlace:  <https://opendata-ajuntament.barcelona.cat/> |
| https://opendata.ecdc.europa.eu/covid19/casedistribution/csv | Devuelve toda la información de datos públicos acerca de la COVID-19 en Europa en formato encolumnado | Este endpoint no requiere parámetros de entrada para recoger los datos del CSV | Debido a la gran extensión del formato de salida de los datos, si se desea consultar el formato de salida de los datos de este endpoint acceder al siguiente enlace:  <https://www.ecdc.europa.eu/en/publications-data/download-todays-data-geographic-distribution-covid-19-cases-worldwide> |

# Conceptos teóricos

En este apartado se abordarán aquellos conceptos teóricos y explicaciones necesarias para entender ciertos aspectos del sistema desarrollado con un mayor nivel de comprensión y detalle. Estos conceptos aúnan explicaciones que no son de conocimiento general en el ámbito de la informática y , por ende, se explican a continuación con el fin de tener una base teórica a la hora de explicar conceptos posteriores.

## ETL (Extract, Transform and Load)

Los procesos ETL (extraer, transformar y cargar, de sus siglas en inglés *Extract, Transform and Load*) son una importante parte de la integración de datos, sirviendo como un elemento importante dentro de las arquitecturas del sector de *Big Data* [1]cuya función completa el resultado de todo el desarrollo de la cohesión de aplicaciones y sistemas.

Es el proceso que permite a las organizaciones mover datos desde múltiples fuentes, reformatearlos y limpiarlos, cargarlos en otra base de datos para analizar, o en otro sistema operacional para apoyar un proceso de negocio.

ETL son las siglas de los pasos que conforman el proceso:

* **Extracción:** consiste en obtener datos de una o varias fuentes; estas pueden ser bases de datos, *APIs*, documentación, CSV (valores separados por comas, de sus siglas en inglés *Comma-Separated Value*), etc.
* **Transformación:** en este paso se realizan los cálculos, validaciones y limpieza de los datos obtenidos de la extracción, adecuándolos con el objetivo de obtener la información necesaria.
* **Carga:** los datos, una vez modificados para la obtención de información, se vuelcan en almacenes de datos, que al igual que en la extracción, puede ser de diferentes tipos: ficheros, bases de datos, *Data Warehouse,* etc.

Un ejemplo de la esquematización de un proceso *ETL* se puede observar en la Ilustración 1.

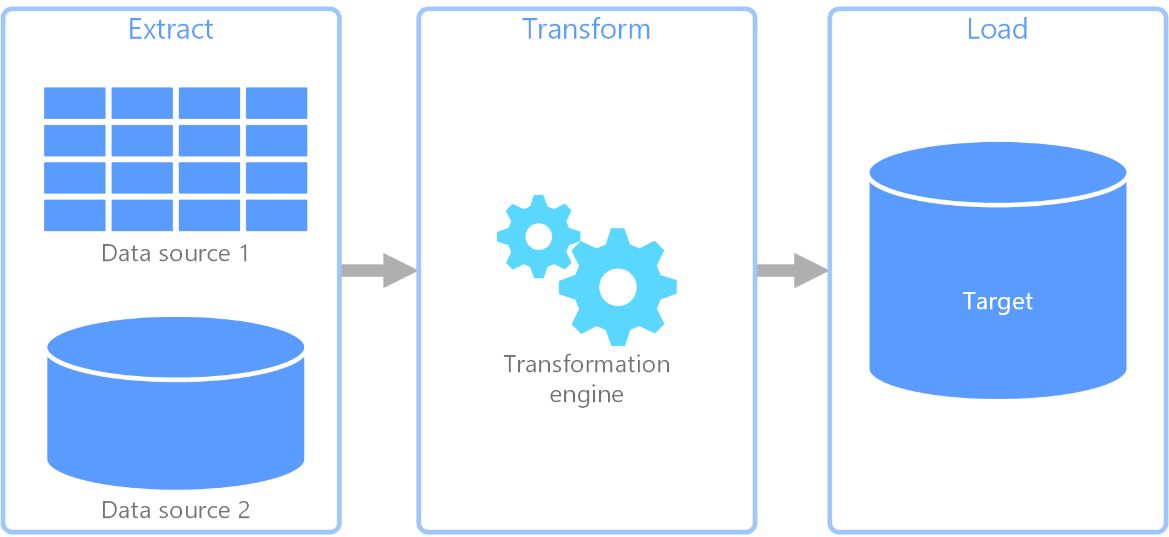


Ilustración : Esquema ETL

Este proceso ha sido utilizado en el proyecto debido a que permite obtener una gran cantidad de información de diversos medios, transformarla, adaptarla y almacenarla de forma que se pueda hacer uso posteriormente de estos datos [2].

## Web Scraping

El concepto de *web scraping* basa su definición en dos pasos principales; la “búsqueda y descarga sistemática y automática de páginas webs y la extracción de información y contenido de estas”. Cada vez que se mencione una fuente de datos, se estará haciendo referencia a la página webs o portal web de empleo de la que se extraerá la información.

El propósito de un *scraper* es la extracción de datos de las páginas webs a través de la “descarga” del contenido HTML (Hypertext Transfer Protocol) de la página web. Este proceso, normalmente automático, involucra la descarga del contenido de una página web (proceso realizado por un buscador cuando una página web es mostrada). Estas páginas webs pueden ser obtenidas como resultado del proceso de *web crawling*, para, posteriormente a su descarga, extraer la información adecuada para ser analizada, formateada, procesada o almacenada [3].

Las páginas web se crean utilizando lenguajes de marcado de texto (HTML y XHTML) y a menudo contienen una gran cantidad de datos útiles como texto. Sin embargo, la mayoría de las páginas web están diseñadas para usuarios finales humanos y no para facilitar el uso automatizado.

El uso particular de la técnica de *web scraping* es usada con propósitos de indexación, búsqueda web o *data mining*, entre otros.

## NLP

También conocido por sus siglas *Natural Language Processing* (NLP) o Procesamiento del Lenguaje Natural, se trata de la práctica de comprensión acerca de cómo las personas organizan sus pensamientos, sentimientos, lenguaje y comportamiento para producir los resultados que obtienen. Este método proporciona a las personas una metodología para modelar un desarrollo personal y para el éxito de los negocios.

Un elemento clave de este campo de las ciencias de la computación, muy utilizado también en el campo de la lingüística y la inteligencia artificial, son las interacciones entre un ordenador y su capacidad de comprensión del lenguaje humano.

Esta práctica toma por referencia el lenguaje humano para aprender modelos computacionales útiles para la realización de determinadas tareas. Este campo cubre otros muchos conceptos dentro del procesamiento de lenguaje humano.

Un ejemplo de tipo de procesamiento de lenguaje natural sería el de NER o reconocimiento de entidades nombradas dentro de un texto. En el que, a través del aprendizaje y enriquecimiento del lenguaje humano, un sistema es capaz de detectar entidades relevantes o destacadas dentro de un texto con el fin de realizar análisis o estadísticas de clasificación más avanzadas.

Otro sería el campo de NLU o entendimiento del lenguaje natural. El cual, gracias a este concepto, permite valorar, identificar y clasificar un texto o una frase en base a su aproximación o cercanía a la forma de expresión de otros textos similares.

## Algoritmo LDA

A partir del llamado *topic modelling* se consigue un proceso de identificar temas en un conjunto de documentos. Esto puede ser útil para los motores de búsqueda, la automatización del servicio al cliente y cualquier otra instancia en la que conocer los temas de los documentos sea importante. Hay varios métodos para hacer esto, a continuación, se explicará uno de ellos: Latent Dirichlet Assignation (LDA).

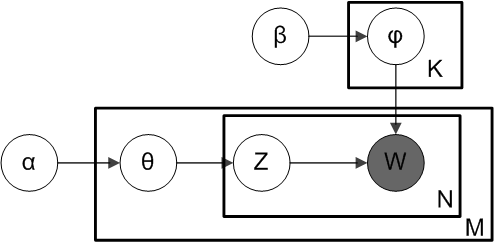
LDA es un algoritmo de aprendizaje no supervisado que ve los documentos como bolsas de palabras o *bags of words* (es decir, el orden no importa). LDA funciona primero haciendo una suposición clave: la forma en que se generó un documento fue seleccionando un conjunto de temas y luego para cada tema eligiendo un conjunto de palabras. Para encontrar estos temas la respuesta se halla en aplicar ingeniería inversa a este proceso **REFERENCIA: https://towardsdatascience.com/lda-topic-modeling-an-explanation-e184c90aadcd**. Para ello, los pasos a realizar para cada documento ‘m’ es el siguiente:

* Suponer que hay un número ‘k’ de temas en todos los documentos
* A continuación, se distribuyen estos ‘k’ temas en el documento ‘m’ (esta distribución se conoce como α y puede ser simétrica o asimétrica) asignando a cada palabra un tema.
* Para cada palabra ‘w’ en el documento ‘m’, hay que suponer que el tema asignado es incorrecto, y que al resto de las demás palabras se les asigna el tema correcto.
* Asignar probablemente la palabra ‘w’ a un tema basado en dos cosas:
  + ¿qué temas hay en el documento ‘m’?
  + ¿cuántas veces a la palabra ‘w’ se le ha asignado un tema en particular en todos los documentos ¿(esta distribución se llama β)
* Finalmente, basta con repetir este proceso varias veces para cada documento y se formaría un conjunto o “bolsa” de palabras con los temas recogidos.

En cuanto a las variables que entran en funcionamiento en el algoritmo destacar las anteriormente mencionadas. Por un lado, α es una matriz donde cada fila es un documento y cada columna representa un tema. Un valor en la fila i y la columna j representa la probabilidad de que el documento i contenga el tema j. Una distribución simétrica significaría que cada tema se distribuye uniformemente en todo el documento, mientras que una distribución asimétrica favorece ciertos temas sobre otros. Esto afecta el punto de partida del modelo y se puede utilizar cuando tenga una idea aproximada de cómo se distribuyen los temas para mejorar los resultados.

Por otro lado, β es una matriz donde cada fila representa un tema y cada columna representa una palabra. Un valor en la fila i y la columna j representa la probabilidad de que el tema i contenga la palabra j. Por lo general, cada palabra se distribuye uniformemente a lo largo del tema, de modo que ningún tema esté sesgado hacia ciertas palabras. Sin embargo, esto puede aprovecharse para sesgar ciertos temas y favorecer ciertas palabras. Por ejemplo, si sabe que tiene un tema sobre los productos de Apple, puede ser útil sesgar palabras como "iphone" y "ipad" para uno de los temas con el fin de impulsar el modelo hacia la búsqueda de ese tema en particular.

A continuación, en la, se muestra la imagen del modelo representado de forma más grafica del funcionamiento del algoritmo



# Tecnologías y herramientas utilizadas

En este apartado se explicarán y detallarán aquellas herramientas y tecnologías utilizadas para el desarrollo de la aplicación y sus subsiguientes subsistemas, dando trasfondo y orientación de los conceptos explicados anteriormente y con una comprensión más detallada de forma individual de cada uno de los componentes de la arquitectura inicialmente planteada.

## Lenguajes de programación

En este apartado se definirán y explicarán brevemente los lenguajes de programación utilizados en el desarrollo de la aplicación tanto de la parte de desarrollo web o *frontend* como de la parte de lógica de negocio o gestión de las APIs, *backend*.

### Python

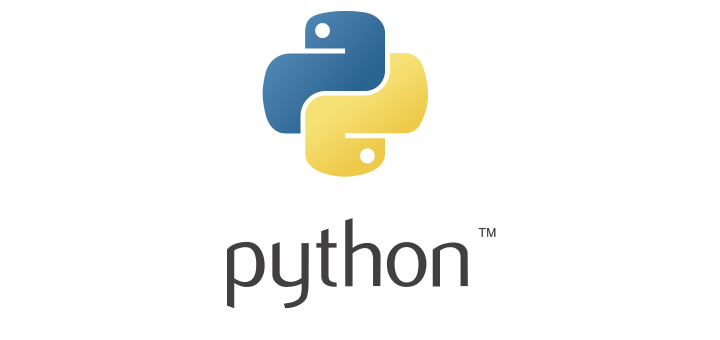


Ilustración : Logo de Python

Es un lenguaje **versátil multiplataforma y multiparadigma** que se destaca por su código legible y limpio. Es el principal lenguaje de programación utilizado en la parte de *backend* del proyecto, más concretamente, en el desarrollo del proceso ETL.

Python es ideal para trabajar con **grandes volúmenes de datos** ya que, el ser multiplataforma, favorece su extracción y procesamiento, es por esto una de las razones por las que se ha utilizado en este proyecto, con una parte dedicada al campo de *Big Data,* para la gestión de los volúmenes de datos extraídos acerca de ofertas de empleo y formación de los distintos portales web.

En el desarrollo de esta aplicación un factor importante debido a la implementación y ejecución de varios módulos dentro del proceso de extracción de datos. Los cuales, en cualquier otro lenguaje, supondrían un consumo de tiempo y de código programado mayor.

Entre algunas de sus múltiples ventajas, a continuación, se encuentran de manera resumida un conjunto de ellas [4]

* **Simplificado y rápido**: Este lenguaje simplifica mucho la programación, es un gran lenguaje para *scripting*.
* **Elegante y flexible**: El lenguaje ofrece muchas facilidades al programador al ser fácilmente legible e interpretable. Permite el tipado “dinámico” de variables, permitiendo declarar con seguridad una variable sin conocer su tipo de datos. Aunque esto, en ocasiones, puede resultar una desventaja si el código no está bien documentado.
* **Ordenado y limpio**: es muy legible y sus módulos están bien organizados.
* **Portable**: Es un lenguaje muy portable. Se puede usar en prácticamente cualquier sistema de la actualidad.
* **Comunidad**: Cuenta con un gran número de usuarios. Su comunidad participa activamente en el desarrollo del lenguaje.

### VueJS

### Node.js



Ilustración : Logo de Node.js

A diferencia de Python, el otro lenguaje de programación importante que se ha utilizado en la aplicación es Node.js, un entorno de ejecución en JavaScript especializado en aplicaciones webs de tiempo real, servidores webs y en el manejo de recursos computacionales a alto nivel. Está basado en el motor JavaScriptV8 de Google. Este motor está diseñado para ejecutarse en un navegador y ejecutar código JavaScript extremadamente rápido. La tecnología detrás de Node.js permite que este motor funcione en el lado del servidor, abriendo una gama completamente nueva de posibilidades cuando se trata del mundo del desarrollo [5].

Para trabajar de manera óptima, Node.js delega todo el trabajo a un grupo de subprocesos. Esta biblioteca tiene su propio entorno asincrónico de subprocesos múltiples. Node.js envía el trabajo a realizar al grupo.Node.js se diseñó teniendo en cuenta la escalabilidad, en particular con la capacidad de admitir una gran cantidad de conexiones simultáneas a un servidor [6]. Muchas tecnologías del lado del servidor ejecutan el entorno para cada una de las solicitudes en un hilo separado. Cuando aumenta el número de solicitudes, aumentan los recursos consumidos en el servidor. Además de los factores determinantes para el rendimiento de una computadora (RAM, CPU, velocidad de conexión), en un servidor las muchas veces el cuello de botella son los procesos de entradas y salidas (E/S).

Esto permite que, en el desarrollo de la aplicación web del proyecto actual, Node.js pueda manejar múltiples conexiones y solicitudes de manera muy eficiente, permitiendo soportar una cantidad considerable de conexiones simultáneas. Por el contrario, en general, Node.js no es adecuado en aplicaciones que requieren un número reducido de conexiones con un alto consumo de recursos (aplicaciones de cálculo, acceso intensivo a datos, etc.).

### TypeScript

## VaderSentiment

## Scikit-learn

Scikit-learn es una de las librerías gratuitas que ofrece Python dentro del área de Data Science o ciencia de los datos para la composición y uso de algoritmos de clasificación, regresión, *clustering* y reducción de dimensionalidad, presentando la compatibilidad con otras librerías de Python como NumPy, SciPy y matplotlib.

La gran variedad de algoritmos y utilidades de Scikit-learn la convierten en la herramienta básica para empezar a programar y estructurar los sistemas de análisis datos y modelado estadístico. Estos se combinan y depuran con otras estructuras de datos y aplicaciones externas como Pandas. REFERENCIA: https://www.master-data-scientist.com/scikit-learn-data-science/

Esta biblioteca se ha utilizado principalmente para la composición de un modelo mediante el algoritmo LDA, el cual se ha explicado anteriormente, que ha permitido la extracción de *topics* o temas de las noticias extraídas sobre la COVID-19 , con los cuales filtrar en la búsqueda de la API de Twitter aquellos tweets que hablasen del virus y de cada uno de los temas o *topics* extraídos.

## PM2

Se trata de un gestor de procesos *daemon* del sistema de GNU/Linux que permite manejar y gestionar de forma sencilla aplicaciones y servicios a través de una interfaz CLI(siglas en inglés de *Command-Line Interface*) [7]. Sirve principalmente para lanzar aplicaciones de Node.js en segundo plano, aunque también sirve para lanzar procesos realizando otro tipo de tareas como la ejecución de *scripts* o código en otros lenguajes de programación como Python.

Se trata de una librería gratuita que se suele utilizar en el desarrollo de aplicaciones web capaz de aguantar cantidades enormes de tráfico con un consumo de recursos realmente reducido y con herramientas que permiten realizar la monitorización de las aplicaciones de manera remota [8].

A través de PM2 se puede controlar un conjunto de procesos listados, que se arrancarán nuevamente en caso de error, manteniéndose encendidos mientras la máquina permanezca encendida. Es decir, en el caso que uno de ellos se termine por cualquier motivo, si se lanza una excepción de error en el programa que haga que el proceso se acabe, PM2 lo iniciará de nuevo automáticamente.

También permite otras herramientas como la gestión de logs, las herramientas de monitorización, el proceso de observación de archivos para rearranque automático cuando el código de la aplicación cambia, las utilidades de despliegue mediante un fichero JSON, etc.

## Amcharts

# Funcionamiento del sistema

En este apartado se mostrará y explicará , mediante una serie de capturas, el funcionamiento de la aplicación web y del sistema planteado de la solución final del proyecto Twico, con el fin de que sirva como guía de utilidad y como manual principal para los usuarios ajenos a la aplicación que deseen proveer de sus servicios.

# Bibliografía

|  |  |
| --- | --- |
| [1] | O. B. Z. A. M. Bala, «Big-ETL: Extracting-Transforming-Loading Approach for Big Data,» Algeria, 2015. |
| [2] | S. Pearlman, «Talend,» 19 Agosto 2019. [En línea]. Available: https://es.talend.com/resources/what-is-etl/. [Último acceso: 2020]. |
| [3] | D. K. M. a. L. Singh, "A dive into Web Scraper world," 2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi: INDIACom, 2016, pp. 689-693. |
| [4] | «Ventajas y desventajas de Python,» Covantec, 2018. [En línea]. Available: https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion1/ventajas\_desventajas.html. [Último acceso: 2020]. |
| [5] | «Documentación de Node.js,» Node.js, [En línea]. Available: https://nodejs.org/es/docs/. [Último acceso: 2020]. |
| [6] | «Node.js: ¿Qué es y para que sirve NodeJS?,» Apasionados del marketing, 30 Septiembre 2015. [En línea]. Available: https://apasionados.es/blog/nodejs-4430/. |
| [7] | «PM2,» PM2, [En línea]. Available: https://pm2.keymetrics.io/docs/usage/pm2-doc-single-page/. [Último acceso: 2020]. |
| [8] | «Ejecutar una aplicación NodeJS en producción con PM2,» DesarrolloWeb, 6 Febrero 2020. [En línea]. Available: https://desarrolloweb.com/articulos/ejecutar-aplicacion-nodejs-pm2.html. [Último acceso: 2020]. |