Capítulo 3

INFRAESTRUCTURA UTILIZADA

En este capítulo se describen las diferentes tecnologías web, de bases de datos y de visualización que se han utilizado en el transcurso del proyecto.

3.1. Tecnologías Web

El desarrollo de sitios web o aplicaciones *online* es un ámbito muy amplio con un gran abanico de tecnologías a disposición de los desarrolladores. En esta sección se explorarán las tecnologías web envueltas en el desarrollo del proyecto tanto en el lado cliente como servidor.

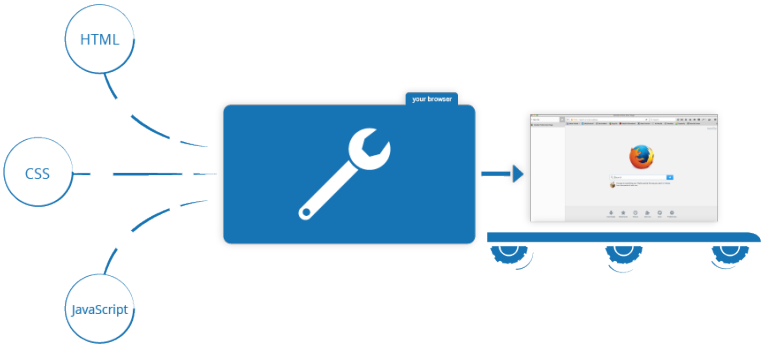

Figura 3.1: Tecnologías web básicas.

3.1.1. HTML

HTML (Hipertextual Markup Lenguaje) [1], es un lenguaje de marcado. Actualmente utilizado para la definición de estructura básica de los contenidos de una página web como vídeos, gráficos, texto...

Publicado en 1991, su historia se remonta a 1980, cuando Tim Berners-Lee propuso un nuevo sistema para compartir ficheros. Actualmente se ha impuesto como el lenguaje estándar para dar formato a documentos, definido por el *World Wide Web Consortium* (W3C), el cual ha ido evolucionando versión a versión adoptando todas las nuevas exigencias que ofrecen las webs actuales, tanto en el campo de los recursos multimedia como en el de interactividad.

HTML se desarrolla haciendo uso de una estructura de etiquetas o *tags*, dentro de las cuales se pueden incluir cada uno de los elementos que conforman una página web. Dispone de cierta capacidad para aportar estilo y lógica pero estas generalmente se delegan en CSS y JavaScript respectivamente.

La última versión oficial es HTML5, la cual proporciona soporte nativo para audio y vídeo, inclusión de la etiqueta `canvas` usada para generar gráficos y efectos tanto en 2D como en 3D, entre otras mejoras. Es la versión usada en este proyecto.

3.1.2. CSS

CSS (Cascade Style Sheet), es un lenguaje de reglas en cascada utilizado para dotar de diseño gráfico a elementos de una página web. Define, como se mencionó anteriormente, la estética de un documento HTML y por lo tanto de una página Web.

Permite crear webs atractivas y responsivas, que se adapten al dispositivo en que están siendo vistas, ya sea por ejemplo, tabletas, ordenadores o móviles.

Permite mover todas las reglas de estilo (tamaños de fuente o de imágenes, colores, responsividad de elementos a ciertas resoluciones...) a ficheros `*.css`, evitando así redundancia en un mismo documento `*.html`, mejorando así la modularidad e independencia dentro de un proyecto.

La última versión es CSS3, añade novedades como soporte nativo para transiciones y animaciones. Además de herramientas para una maquetación más precisa. Es la versión usada en este proyecto.

3.1.3. JavaScript

JavaScript [2] es un lenguaje de programación ligero, interpretado, orientado a objetos y dinámico. Utilizado principalmente como lenguaje de *scripting* para paginas web, en este campo su papel principal se centra en el

desarrollo de lógica en la parte del cliente *backend*: acceso al *Document Object Model*(DOM) de la web, modificación de etiquetas HTML, generación de gráficos en *canvas* o gestión de *cookies*.

Permite crear nuevo contenido dinámico, así como controlar archivos multimedia y gracias al uso de API's (Aplication Programming Interface), enriquece a JavaScript con más funcionalidades permitiendo crear funcionalidades más complejas con desarrollos más simples.

3.1.4. Python

Python es un lenguaje de programación interpretado, orientado a objetos y de alto nivel. Diseñado para un desarrollo de aplicaciones rápido, se utiliza como lenguaje de *scripting* y conexión entre otros componentes de un sistema.

Python es simple, con una sintaxis fácil de aprender centrada en la legibilidad del código, consiguiendo así reducir el coste del desarrollo, mantenimiento y ampliación de proyectos.

Tiene una gran biblioteca de librerías que puede ser fácilmente extendida por módulos personalizados escritos en C/Python. Haciendo uso del instalador de paquetes `PIP` (Python Package Installer), es posible la instalación e integración de paquetería en proyectos de manera muy sencilla, así como el cambio de versiones de las mismas.

Pese a no ser estrictamenta hablando una tecnología web, en este proyecto lo hemos empleado en el contexto de Django, para programar la lógica *backend*. El proyecto comenzó a desarrollarse en Python 2.6 y ha terminado en la versión Python 3.6.9.

3.1.5. Django

Django es un entorno Web de alto nivel diseñado para desarrollar servidores en Python. Al igual que este, su filosofía se centra en desarrollos rápidos, limpios y en un diseño pragmático. Sigue el patrón *Model-View-Template* (MVT) [3], donde:

- *Model*, esta capa del patrón tiene toda la información relativa a las bases de datos: cómo se almacenan, cómo se relacionan entre ellas, cómo validarlas... Manejado por la capa de bases de datos de Django. Toda esta información de configuración se desarrolla y almacena en el fichero `Models.py`. Mediante su ORM (Object Relational Model) permite al desarrollador del servidor hacer consultas y manejar los datos de bases

de datos relacionales en Python. Los filtros, desarrollados en Python, se traducen automáticamente peticiones `SQL`.

- *View*, parte lógica del Framework, se puede ver como una unión entre la capa de modelo y plantillas. Formado por dos ficheros: `urls.py`, encargado de llamar a la vista adecuada dependiendo de la URL a la que se acceda y `views.py` con todas esas vistas que devolverán una respuesta HTTP y en las cuales se consultará la capa Model si fuese necesario.

- *Template*, sección que se encargará del qué y cómo mostrar los datos. Manejado por vistas y plantillas de Django que servirán de bases para la parte *frontend* de la Web. Guardado en documentos HTML enriquecidos junto a variables de plantillas Django ({{ nombre_de_variable }}), las cuales permite el uso de, por ejemplo, bucles, operaciones condicionales, diccionarios, inserción de bloques... para generar webs enriquecidas y dinámicas en muy pocas líneas de código.

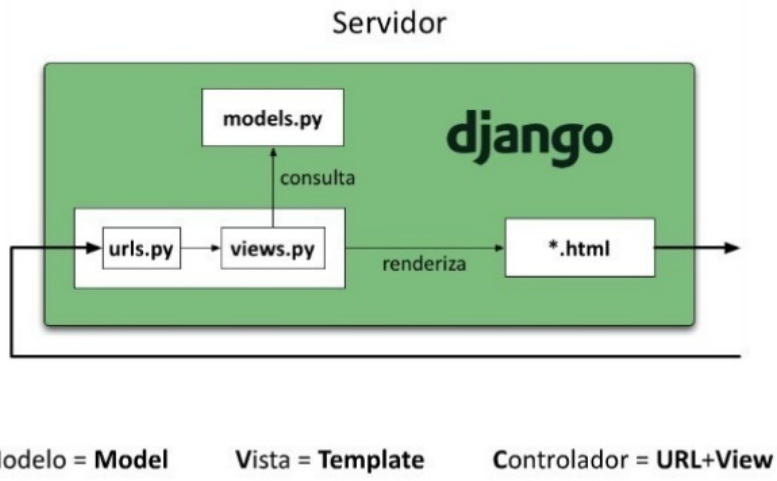

Figura 3.2: Patrón MVT Django.

Diseñado para ser seguro, y evitar errores comunes, Django ofrece una fácil administración de las cuentas de usuarios, así como de sus contraseñas. Además de una sección de administración en la que poder gestionar toda la información almacenada en las bases de datos desplegadas.

El software ya existente de Kibotics Webserver, servicio web sobre el que se ha trabajado, ya estaba basado en Django. El proyecto comenzó a desarrollarse en la versión 1.9 y ha concluido en la Django 1.11 [4].

3.2. Bases de datos

Una base de datos es un conjunto de información, la cual pertenece a un mismo contexto, almacenada de modo sistemático y preservada en distintos formatos, puede ser consultada o alterada posteriormente. En esta sección, se detallarán las distintas tecnologías de bases de datos utilizadas en el desarrollo de este proyecto.

3.2.1. SQLite

SQLite es una librería ligera, rápida y fiable desarrollada en C. Siendo actualmente el motor de bases de datos SQL más usado en el mundo, utilizado en gran parte de los dispositivos móviles y ordenadores, además de venir de serie en muchas aplicaciones, por ejemplo, Django.

No necesita de un servidor para funcionar, hecho por el cual su integración y despliegue es sencillo. Está basado en lectura y escritura en un fichero `*.sqlite` para almacenar toda la información de una base de datos. Este fichero es multiplataforma pudiendo así ser migrado entre distintos sistemas de manera muy sencilla, y tiene un tamaño máximo de 140 terabytes.

Versión? para qué nosotros? URL?

3.2.2. MongoDB

MongoDB [5] es una base de datos NoSQL (Not only SQL) distribuida, documental (almacenando la información en ficheros BSON, muy similares a JSON), de código abierto y diseñada para ofrecer un nivel productivo alto.

Al ser una base de datos NoSQL, permite almacenar información de forma estructurada, pero flexible, pues los esquemas pueden ser cambiados sin necesidad de parar el servicio.

Debido a esta estructura, la velocidad en las consultas es muy alta, convirtiéndose así en una base de datos ideal para trabajar con grandes cantidades de información que vayan a ser consultados muy frecuentemente.

La escalabilidad de MongoDB es sencilla, puesto que se ejecuta en *clusters*, podrá escalar horizontalmente contratando más máquinas, aumentando así la capacidad de procesamiento. Es una base de datos muy utilizada en la industria [6].

Para el primer prototipo de este proyecto, se ha utilizado la última versión estable de MongoDB, la versión 4.2.6.

3.2.3. Elasticsearch

Elasticsearch [7] es una base de datos. Junto a Logstash y Kibana, los tres proyectos open source, forman el stack ELK.

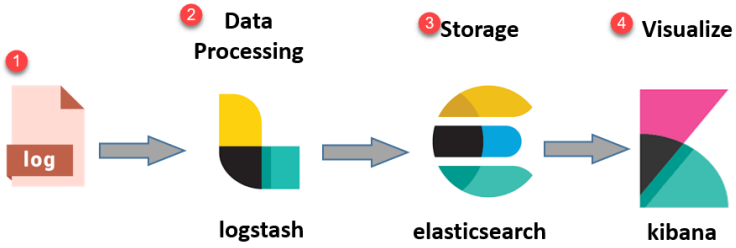

Figura 3.3: Stack ELK.

Basado en Lucene (API para recuperación de información), permite almacenar información compleja como datos de geolocalización así como realizar búsquedas de texto y auto-completado casi en tiempo real. Mediante el uso de una API Rest realiza consultas, borrado y actualización de documentos.

En vez de usar filas y columnas, Elasticsearch almacena estructuras complejas de datos serializadas como documentos JSON. Para almacenar estos documentos, Elasticsearch hace uso de índices. Cada uno de estos índices dispone de una estructura de datos propia. Estas estructuras JSON son tambien utilizadas tanto para las consultas y respuestas, lo que la hace sencilla de usar e integrar en sistemas productivos ya existentes.

Dadas estas capacidades de almacenar información preparada en índices, la consulta de documentos es muy ágil puesto que evita búsquedas tanto en indices como en documentos no deseados. Organizado en nodos, permitirá aumentar la potencia a medida que la demanda de recursos crezca.

Se ha convertido en uno de los buscadores por texto más importantes, utilizado por gigantes de Internet como Facebook, Netflix o Github.

La última versión estable de Elasticsearch es la versión 7.8.0 [8], liberada el 18 de Junio del 2020. En proyecto, se ha utilizando la versión 7.6.2 publicada el 31 de Marzo del 2020.

3.3. Tecnologías de visualización

La muestra de datos es una de las bases de este proyecto, en esta sección se explicarán las tecnologías utilizadas tanto en el primer prototipo como en la versión final del módulo de analíticas desarrollado para este proyecto.

3.3.1. Matplotlib

Matplotlib es una librería de Python que se encarga de la generación de visualizaciones tanto estáticas como animadas.

Proporciona gran variedad de visualizaciones como mapas de calor, gráficas de barras, histogramas... recordando a Matlab. Ofrece cierta capacidad de estilo y puede ser utilizada junto a otras librerías para generar visualizaciones aún más complejas y enriquecidas como mapas geográficos en los que se representa datos mediante latitud y longitud.

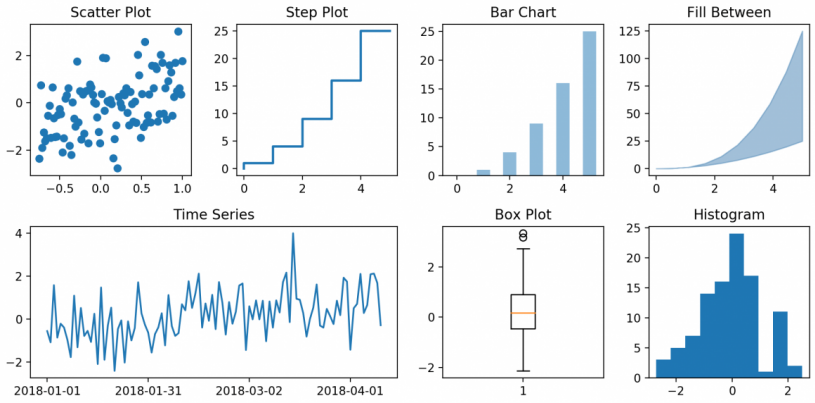

Figura 3.4: Visualizaciones en Matplotlib.

Matplotlib almacena las visualizaciones en figuras, cada una contiene los ejes en que se representarán los datos, estos ejes pueden ser de múltiples tipos, ya sean coordenadas x-y, x-y-x para una representación en tres dimensiones o un eje polar.

Las visualizaciones generadas podrán ser mostradas en una nueva ventana si utilizamos la librería en un *script* o ser renderizadas y devueltas como imagen `PNG` para su posterior muestra en el servicio web haciendo uso de la etiqueta `HTML` <img></img>

En el primer prototipo de este proyecto se utilizó la versión 3.1.2 de Matplotlib, publicada el 18 de Marzo del 2020 [9].

3.3.2. Kibana

Como se comentó anteriormente, el stack ELK está compuesto por Kibana [10] como motor de búsqueda, procesador de datos y generador de visualizaciones entre otras funcionalidades. Diseñada para utilizar Elasticsearch como fuente de datos.

Gracias a su aplicación *frontend*, la creación de visualizaciones se simplifica mucho sin ser necesaria la codificación de estas.

Mediante configuración, filtrado y selección de los datos indexados en Elasticsearch se pueden crear múltiples tipos de visualizaciones interactivas (gráficos de barras, circulares, tablas, histogramas y mapas), y posteriormente ser agrupadas en tableros o *Dashboards*, los cuales permiten la visualización y posterior filtrado de grandes cantidades de información de forma simultanea y sencilla.

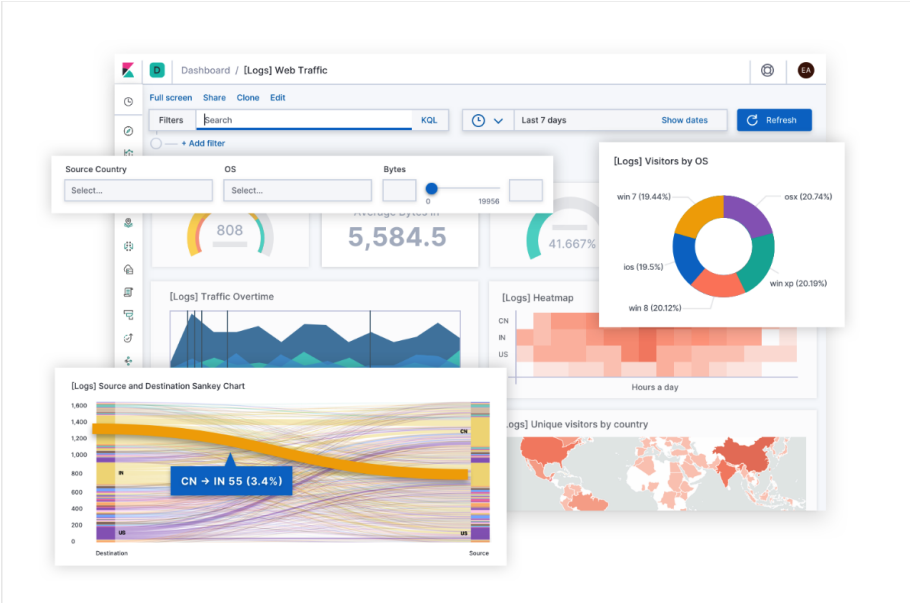

Figura 3.5: Visualizaciones en Kibana.

Permite el procesado de los documentos ya indexados en Elasticsearch para crear nuevos campos dinámicos que podrán ser utilizados y representados posteriormente en visualizaciones y estadísticas.

Ofrece una interfaz segura y organizada, dividida en espacios de trabajo o *spaces*. Estos espacios pueden ser tratados como instalaciones de Kibana independientes permitiendo a varios equipos de trabajo hacer uso del mismo

servicio de Kibana sin los inconvenientes de compartir información de índices. Aumentando la compartimentalización y escalabilidad del servicio.

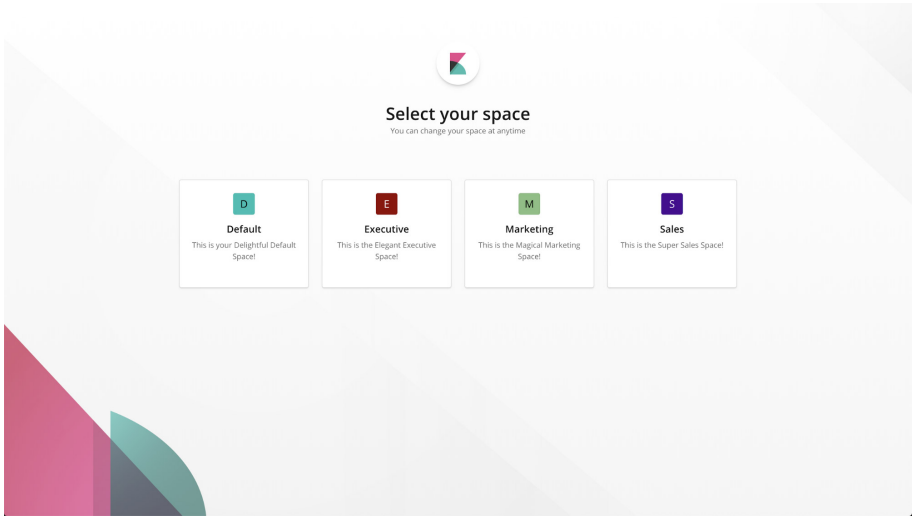

Figura 3.6: Selector de espacios de trabajo en Kibana.

La última versión estable de Kibana es la 7.8.0 [11], liberada el 18 de Junio del 2020. En la versión final desarrollada se ha utilizado Kibana 7.7.0 publicada el 13 de Mayo del 2020.