MDA

DATA PROJECT 2

GRUPO FIJI

Andreu Soriano

Inés Muñoz

Pablo Penadés

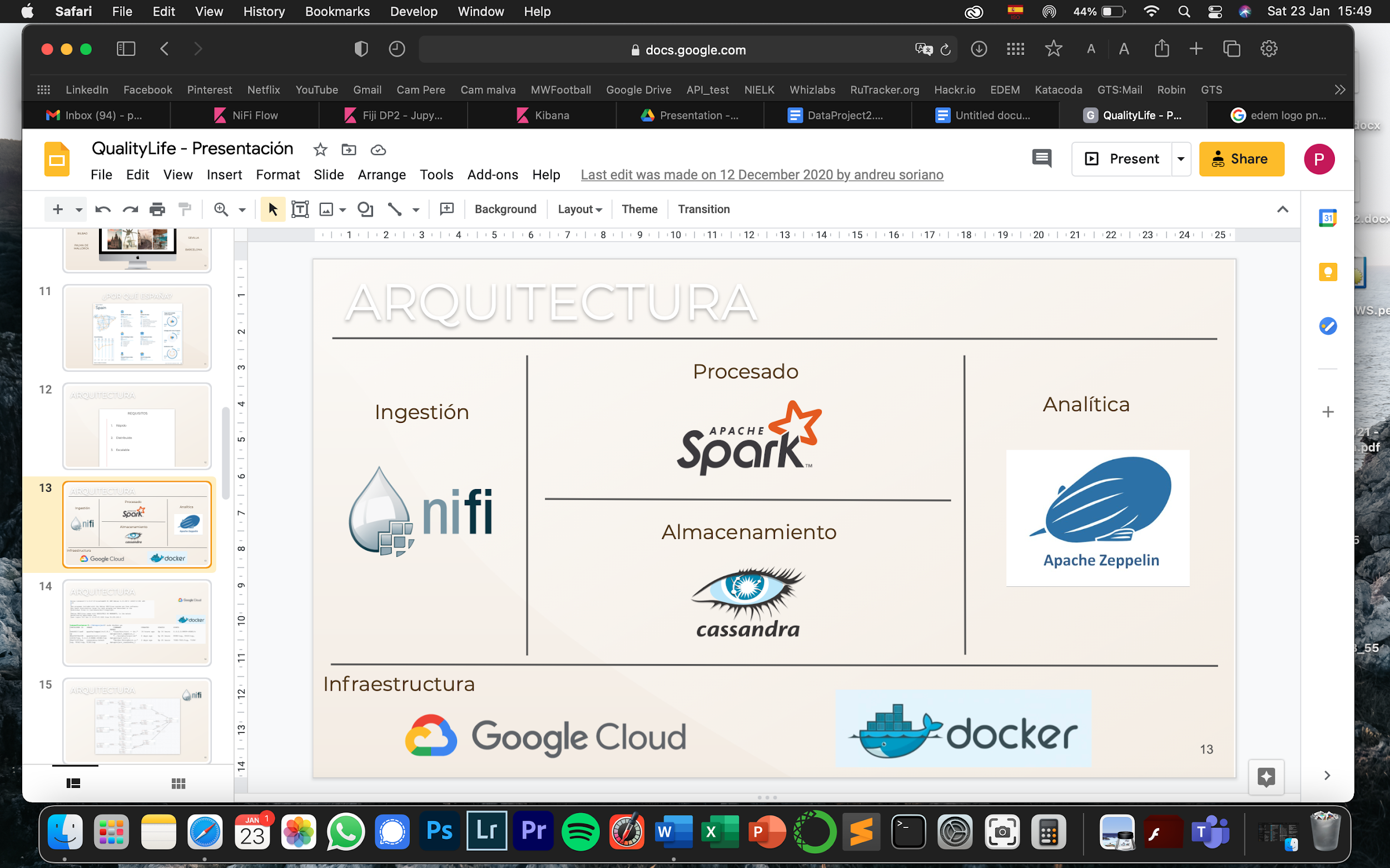
Rosa LLorca

El siguiente documento tiene como objetivo ofrecer una solución al Request For Proposal (RFP) elaborado por QualityLife. A lo largo del documento se detalla un plan de acción, una estimación de costes y la propuesta de un Minimum Viable Product (MVP). Además, se detallan las funcionalidades que enmarcaría la solución completa en respuesta al RFP.

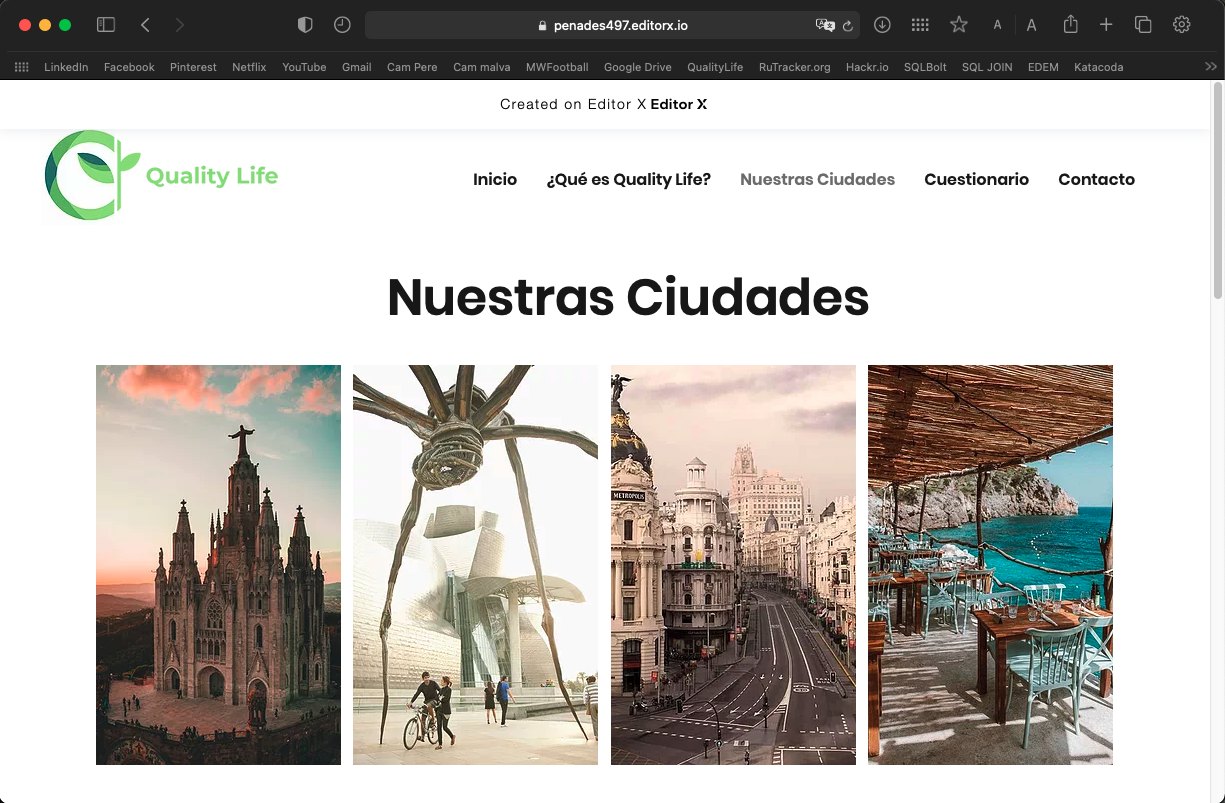
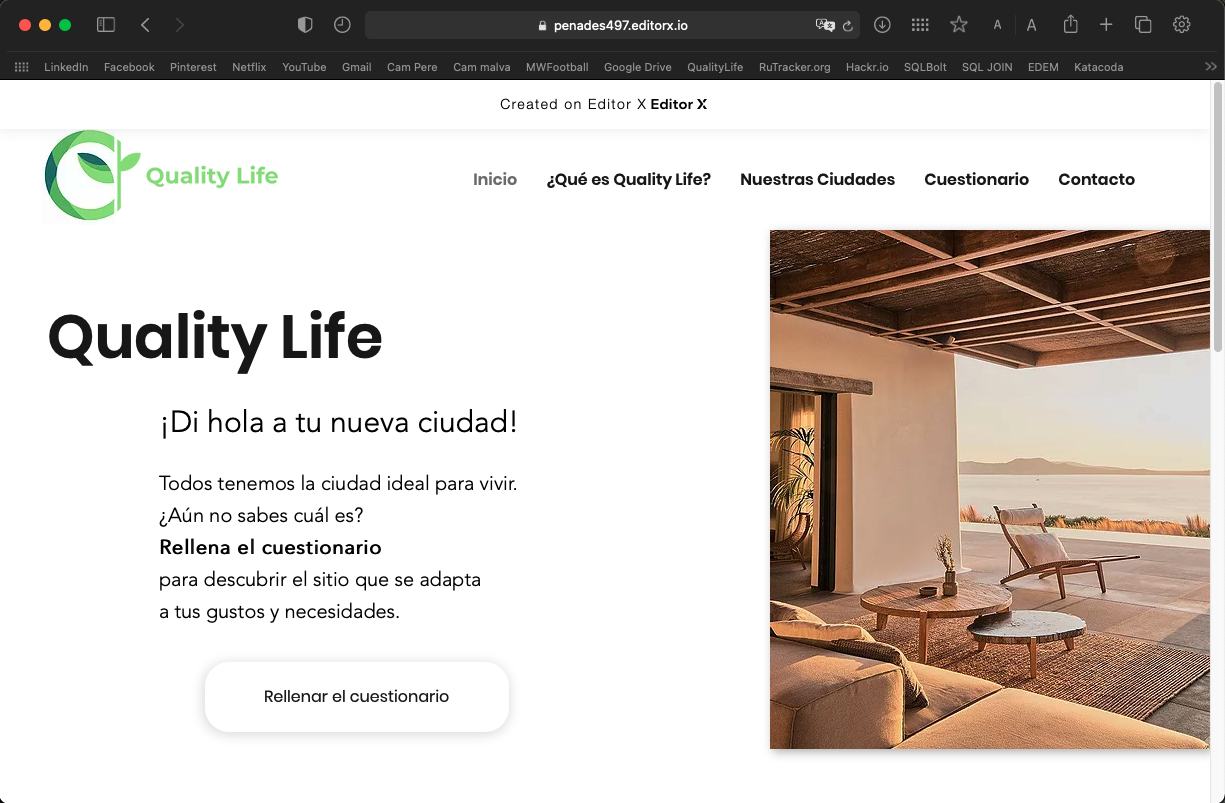
1. CONTEXTO

En un primer proyecto, el Grupo Fiji realizó una prueba de concepto para dar una solución a QualityLife en la cual se ofrecía un servicio para que usuarios de edades maduras y avanzadas pudieran descubrir fácilmente la ciudad que más se acomode a sus gustos y preferencias. El proyecto hizo uso de tecnologías actuales y basadas en la Nube.

*Diagrama 1: Esquema de la arquitectura de software utilizada para el primer proyecto para QualityLife.*



*Imágenes 1 y 2: Capturas de pantalla de la web creada para QualityLife.*



Basados en el target al que quería dirigirse QualityLife, creamos una página web sencilla, clara y elegante para ofrecer una fácil comprensión, uso y atracción por parte de la audiencia. De este modo, creamos una plataforma que redujera la brecha de la compleja arquitectura con la poca digitalización del tipo de cliente deseado.

2. REQUISITOS

Para este nuevo proyecto, QualityLife tiene una serie de requisitos con el objetivo de utilizar otro tipo de canal de comunicación llegando así a una audiencia más amplia. QualityLife quiere utilizar la red social Twitter para ofrecer una de sus propiedades disponibles a cualquier usuario que comunique sus necesidades. En consecuencia, QualityLife pretende dar una solución a un problema de forma personalizada y rápida, ya que el tiempo es un factor determinante a la hora de generar interés acerca de una propiedad. El prototipo a desarrollar debe:

* Leer los tweets de la cuenta de QualityLife y ser capaz de responder.
* Leer utilizando el #mdaedem
* Que la respuesta sea vía Twitter y lo más rápida posible.
* Encontrar el mejor match posible a cada petición.
* Incluir el código de la vivienda así como otros datos de interés opcionalmente.

3. PROPUESTA

El valor de un servicio va más allá del “*core business*” de una empresa. Debe enmarcar una relación integral con el cliente, desde el primer interés generado en el usuario hasta el acompañamiento post-venta. Mudarse es una decisión desafiante, difícil y en muchos casos aterradora. El cliente de Quality Life quiere sentirse acompañado y apoyado, quiere sentirse único y seguro de su decisión. La tecnología puede ayudar a tomar mejores decisiones utilizando datos pertinentes a cada una de las necesidades del cliente.

El Grupo Fiji propone un sistema rápido, eficiente y seguro para ofrecer una respuesta personalizada a cualquier potencial cliente aumentando la tasa de conversión y la confianza del usuario.

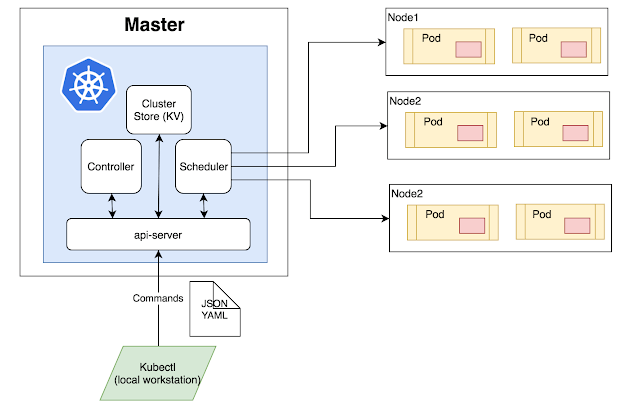
3.1 ARQUITECTURA

DIAGRAMA ARQUITECTURA PRO

3.1.1 INFRAESTRUCTURA (Kunbernetes & Google Cloud)

Kubernetes es la herramienta de orquestación de contenedores de referencia en producción sobre todo en caso de escalado horizontal. Kubernetes permite automatizar procesos de escalado de máquinas virtuales en función de reglas y balancear la carga asignada a cada una de las máquinas. Kubernetes es una herramienta gratuita e incluso ofrecida por proveedores Cloud. En este último caso sí que deriva un coste adicional por las máquinas virtuales. Kubernetes asegurará el adecuado escalado, de ser requerido, cuando el volumen generado por QualityLife sobrepase la capacidad de una de las máquinas virtuales en Cloud.

*Diagrama 2: Funcionamiento de orquestación de Kubernetes.*



En efecto, se propone optar por el procesamiento y almacenamiento en Cloud. En los últimos años, esta opción se ha convertido en cada vez más común debido a la flexibilidad, versatilidad y complementariedad de los servicios en la Nube. Además, una de las grandes ventajas de Cloud es la menor gestión, monitorización y problemas de seguridad en comparación a los servidores en local. Dada la naturaleza del proyecto, orientada a una flexible respuesta en tiempo real de volúmenes y tipos de datos de diversos y amplio rangos, una arquitectura en Cloud es la mejor opción. No solo se responde así al requisito de flexibilidad y escalabilidad sino a un coste más basado en el uso y en definitiva, en la eficiencia de los recursos. Google Cloud Platform es uno de los principales servicios de Cloud a nivel mundial especializado en calidad a precios relativamente asequibles, sobre todo en comparación a los servicios de Amazon o Microsoft. Se opta por Google debido a la fácil migración del entorno del prototipo que se explica posteriormente, a los menores costes de ambos el prototipo y las máquinas una vez puestas en producción y a la experiencia que ya atesora el Grupo Fiji en este proveedor de Cloud.

3.1.2 INGESTIÓN (NIFI)

NIFI es una herramienta de ingestión de datos fácil de usar gracias a su interfaz visual y con una amplia funcionalidad. Una ventaja a destacar de NIFI es su rápida capacidad de ingestión y adaptabilidad.

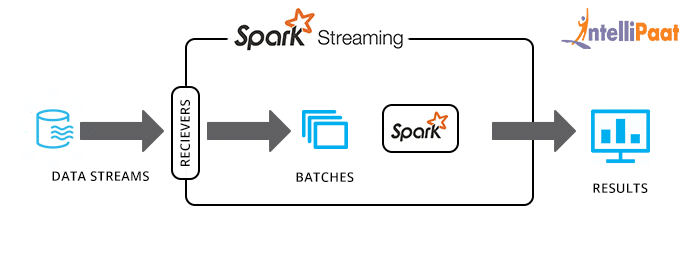
3.1.3 STREAMING (KAFKA)

Kafka es uno de los servicios de streaming más usados y de mayor calidad en el mundo. Esta herramienta se suele utilizar entre la parte de ingestión y de procesamiento para transformar, almacenar y obtener una trazabilidad y organización de los datos enviados para procesar. Kafka está diseñado para comunicar grandes volúmenes de datos con baja latencia para procesos en tiempo-real. Permite dividir y almacenar la información en flujos ordenados por “topic” o temática. Dentro de cada uno de estos topics se puede configurar su persistencia y morfología manteniendo la certidumbre temporal de cada dato gracias al denominado sistema de “offset”.

3.1.4 PROCESAMIENTO (SPARK STREAMING)

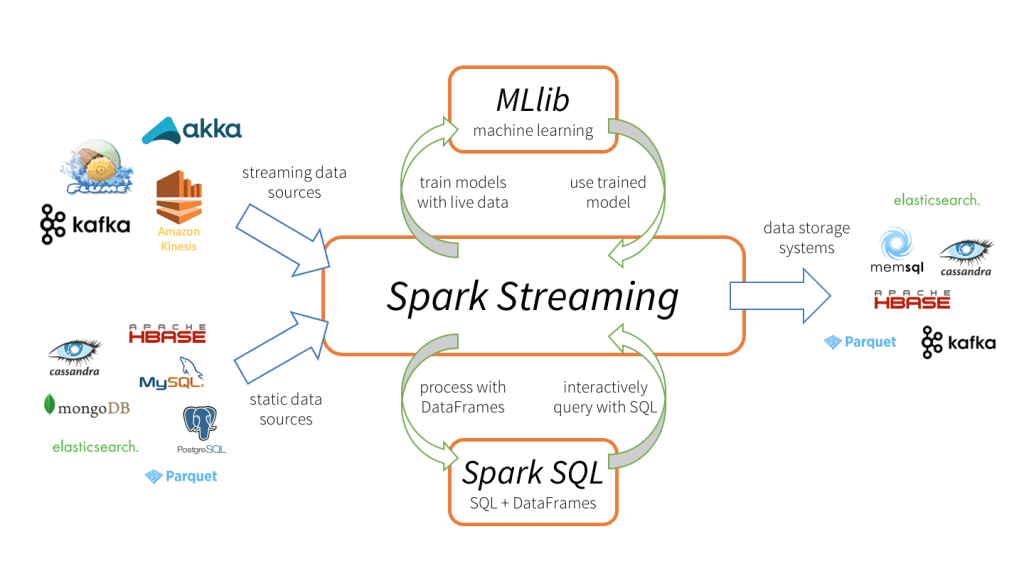
Spark Streaming es una extensión de Spark que provee el procesamiento de flujos de gran volumen en tiempo-real de forma escalable y tolerante al fallo. Spark Streaming se basa en un sistema de micro-batching basado en los flujos continuos de información agrupados en ventanas temporales en constante movimiento. El procesamiento se basa en el core de Spark y por tanto de forma distribuida. QualityLife requiere de un sistema de procesamiento en tiempo real, robusto y escalable. Existen otras herramientas que también ofrecen funcionalidades incluso mejores en cuanto a tiempo real como Flink, no obstante el mayor coste de estas no se justifica ante las necesidades de QualityLife. Teniendo en cuenta que el sistema ha de contestar un tweet rápidamente, el hecho de que sea en menos de 1 minuto no sería un problema y en consecuencia, Spark Streaming es una herramienta que más se adecua en términos de prestaciones y coste.

*Diagrama 3: Funcionamiento conceptual simplificado de Spark Streaming.*



La gran ventaja de Spark streaming es su popular uso en proyectos de procesamiento de gran potencia y por tanto, su conectividad con otras herramientas. De este modo, se puede utilizar Kafka como fuente de datos, ejecutar el procesamiento necesario y enviar los datos a una base de datos como elasticsearch o S3.

*Imagen 3: Ejemplos de conectividad de Spark Streaming con multitud de herramientas.*



3.1.5 ALMACENAMIENTO (ELASTICSEARCH & S3)

Elasticsearch se basa en un sistema de indexación inversa por el cual, se busca en función del contenido en vez de en función del título. Elasticsearch es excelente para labores de búsqueda de ficheros de texto, sobre todo para aquellos con un patrón similar. Esta herramienta ofrece funcionalidades tanto como motor de búsqueda como de base de datos, proporcionando una gran rapidez y versatilidad.

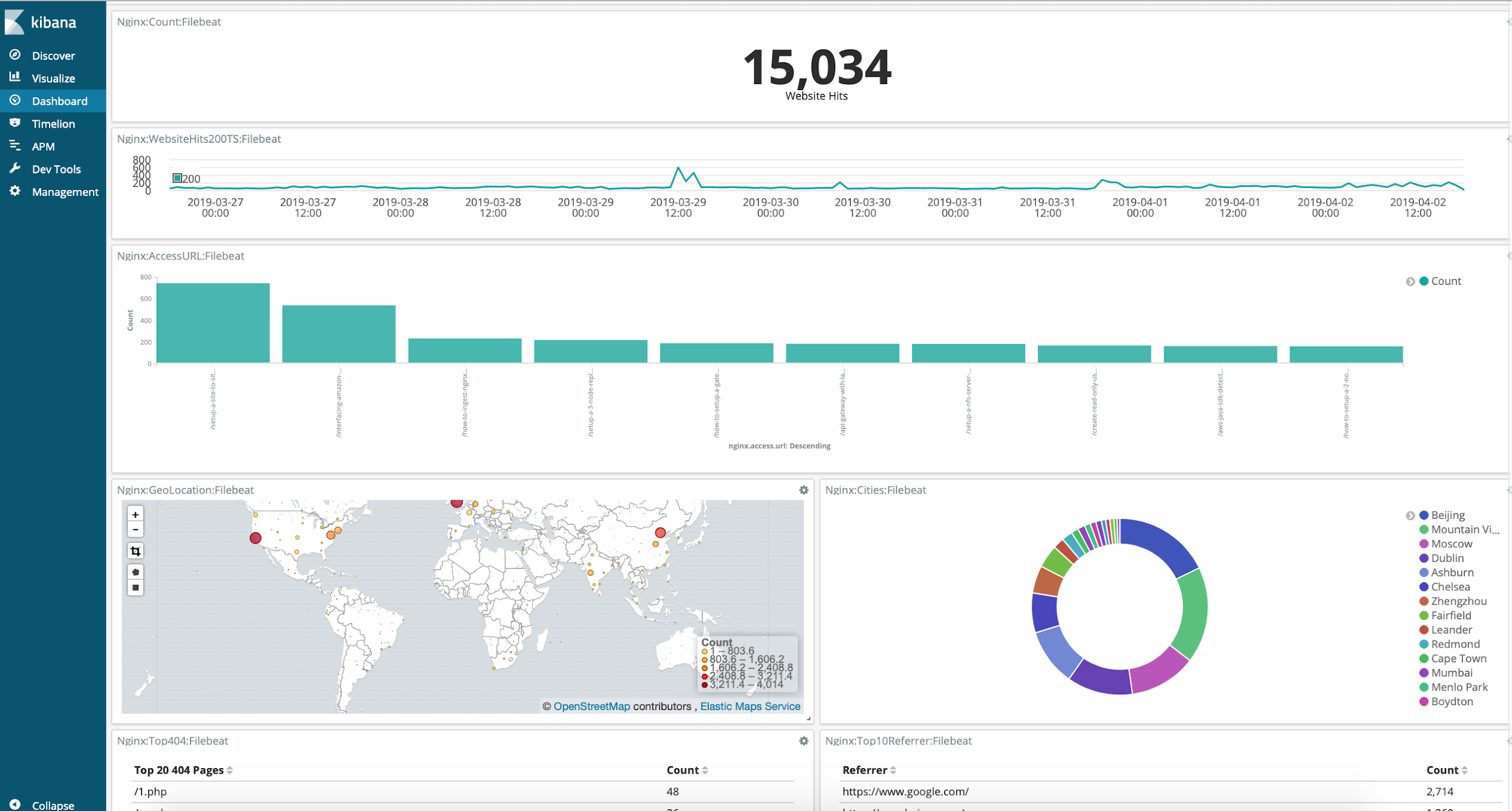
El almacenamiento en Elasticsearch se basa en la creación de índices los cuales almacenan documentos. A raíz de un índice, se puede establecer un patrón gracias al cual se realizan búsquedas a modo query. Este patrón define una estructura clave valor adaptada al contenido y configurable.

Elasticsearch será nuestra base de datos para todo lo relativo a los tweets. No obstante, Quality Life necesitará una base de datos más amplia para almacenar información acerca de las propiedades así como sus fotos, descripciones, comentarios… Además, será necesario almacenar la información de los clientes y los logs del sistema. En función de los distintos tipos de datos a almacenar se puede intuir que algunos serán utilizados con frecuencia y otros no. S3 es el tipo de almacenamiento que se ajusta a estos requisitos. Amazon S3 ofrece una variedad de clases de almacenamiento diseñadas para diferentes casos de uso. Estos incluyen S3 Standard para el almacenamiento de uso general de datos a los que se accede con frecuencia; S3 Intelligent-Tiering para datos con patrones de acceso desconocidos o cambiantes; S3 Standard-Infrequent Access (S3 Standard-IA) y S3 One Zone-Infrequent Access (S3 One Zone-IA) para datos de larga duración, pero a los que se accede con menos frecuencia; y Amazon S3 Glacier (S3 Glacier) y Amazon S3 Glacier Deep Archive (S3 Glacier Deep Archive) para el archivo a largo plazo y la preservación digital. Este método permite abaratar costes en aquella información que es necesaria almacenar a modo de histórico o por motivos legales.

3.1.6 VISUALIZACIÓN (KIBANA)

Kibana es la plataforma de visualización de Elasticsearch donde se pueden crear índices, patrones de índices, visualizar documentos e información en forma de dashboards. Kibana es una herramienta muy versátil para entender patrones y poder visualizarlos de forma sencilla. Ya que la fuente de datos es de tipo JSON, Kibana resultará muy útil para visualizar en tiempo real los requisitos, zonas calientes y cantidades o zonas solicitadas/disponibles entre otros en dashboards.

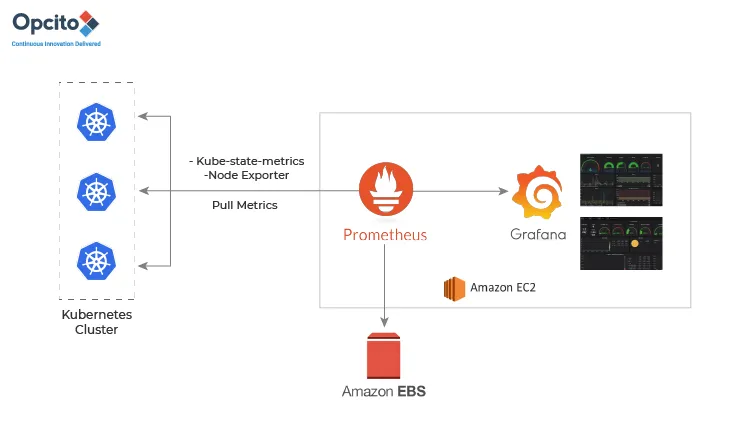
*Imagen 4: Ejemplo de dashboard en Kibana.*



3.1.7 MONITORIZACIÓN (GRAFANA & PROMETHEUS)

Grafa es una de las herramientas más utilizadas en lo que a monitorización de sistemas. Grafana ofrece la visualización del estado del sistema mediante analítica y cuadros de mandos. En paralelo, se pueden crear alertas cuando ciertos niveles son alcanzados para poder prevenir o resolver a tiempo cualquier incidencia en relación a la salud de las herramientas. Prometheus es el sistema de monitorización detrás de Grafana con el cual se pueden realizar queries de gran potencia. Prometheus sirve de punto de enlace con los clusters en Kubernetes.

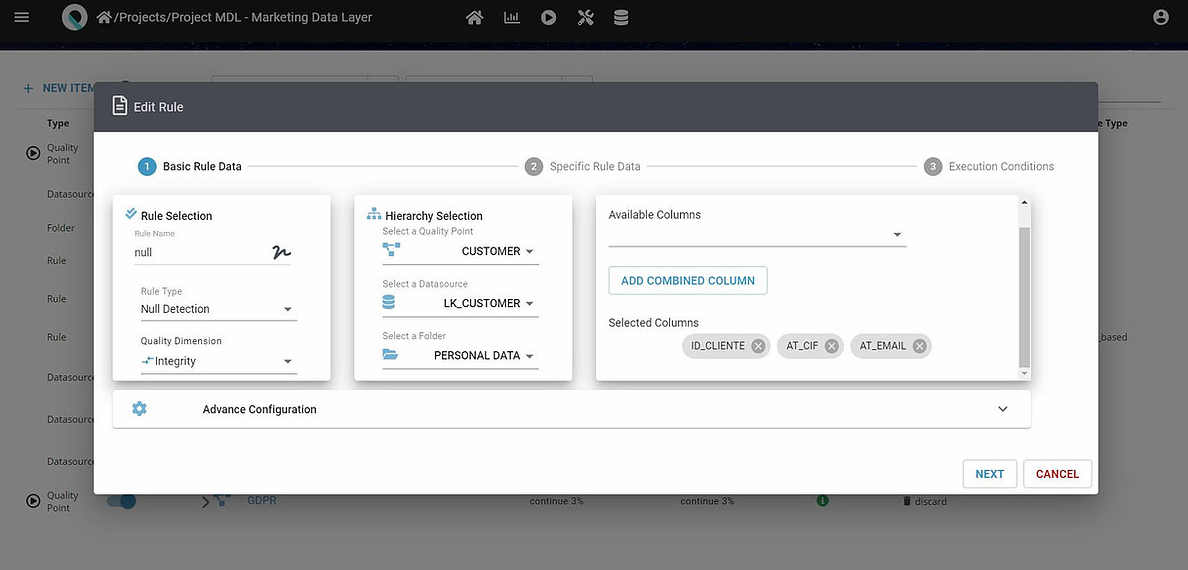
*Diagrama 4: Funcionamiento simplificado de Prometheus & Grafana basado en una infraestructura en AWS.*



3.1.8 CALIDAD DEL DATO (AQTIVA)

Aqtiva es una herramienta de calidad del dato basada en una interfaz gráfica para poder conectar de forma sencilla fuentes de datos y crear reglas de verificación y checks para flujos BigData. La empresa utiliza tecnologías distribuidas basadas en Azure para procesar grandes volúmenes de datos. Aqtiva también ofrece la conexión de los datos tratados a un servicio de almacenamiento propietario en la nube(en este caso S3). El servicio es considerablemente integral y a un precio competitivo teniendo en cuenta los altos precios de otros competidores.

*Imagen 5: Ejemplo de regla en la plataforma Aqtiva.*

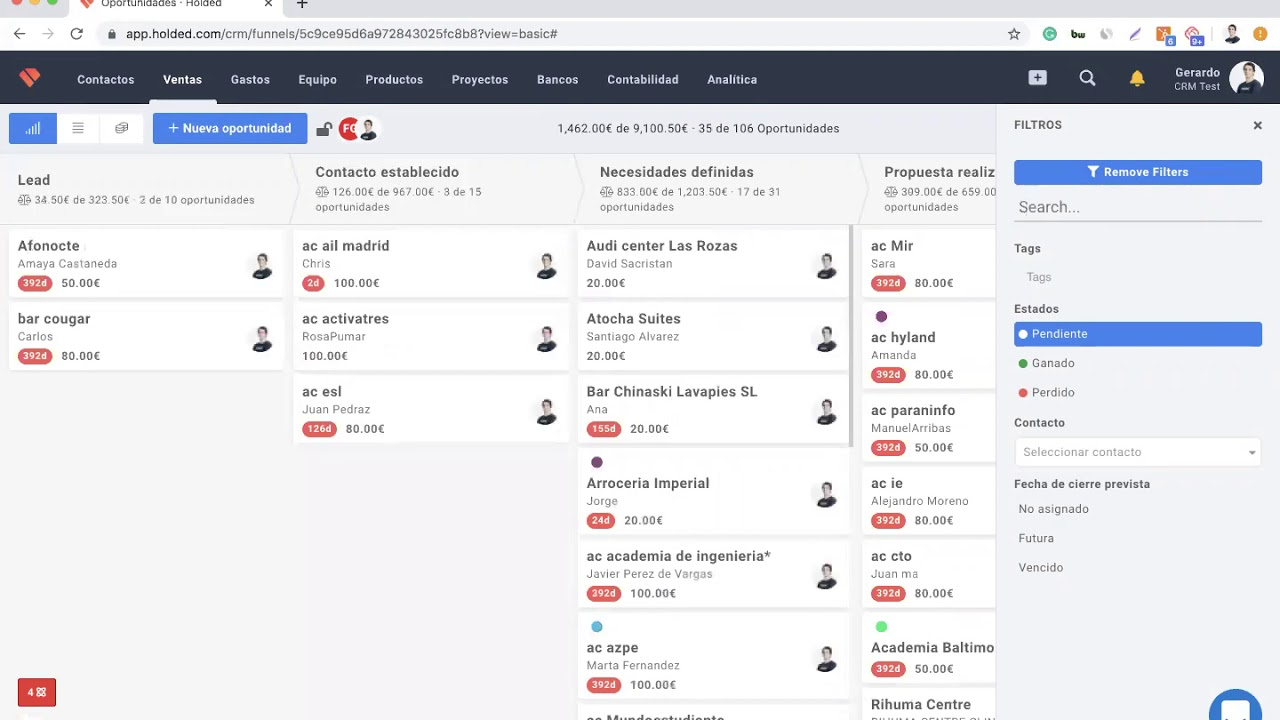


3.2 METODOLOGÍA

Es imprescindible un buen sistema de organización y comunicación en el equipo para mantener los plazos y minimizar riesgos y errores que puedan acarrear sobrecostes. Proponemos la implantación de metodologías Ágiles con plazos de 2 semanas, frecuente comunicación y reporting y tareas desglosadas para un seguimiento y cumplimiento más factibles. Además, debido a la naturaleza de Quality Life, será necesario contar con una herramienta de CRM que permita el seguimiento de los (potenciales) clientes.

De este modo, proponemos el uso de la herramienta HOLDED. Esta herramienta ofrece un servicio integral de gestión empresarial, desde CRM hasta analítica pasando por Finanzas o Equipo entre otros.

*Imagen 6: Ejemplo de CRM HOLDED.*



4. ROLES NECESARIOS (TECNOLOGÍAS Y PERFILES)

El time-to-market es un factor clave para posicionarse en el mercado digital. El mercado inmobiliario engloba una considerable madurez en su vertiente digital donde la imagen de marca es un elemento determinante que suele basarse en la confianza establecida por el usuario. Esperamos poder comenzar el proyecto de forma rápida y eficiente. Durante los últimos años, muchas empresas digitales han optado por subcontratar ciertas fases de sus procesos de desarrollo en países donde los salarios son más bajos como en zonas del Este de Europa, Oriente Medio o Asia Occidental. En muchas ocasiones, es una opción a considerar. Sin embargo, requieren de una mayor planificación y clara distribución del trabajo debido a las barreras geográficas y culturales. Creemos que España, y en concreto Valencia, alberga el suficiente talento como para poder encontrar los perfiles necesarios y empezar cuanto antes sin necesidad de añadir labores extra de coordinación.

Para el desarrollo del proyecto, el Grupo Fiji propone los siguientes puestos necesarios así como su nivel de experiencia:

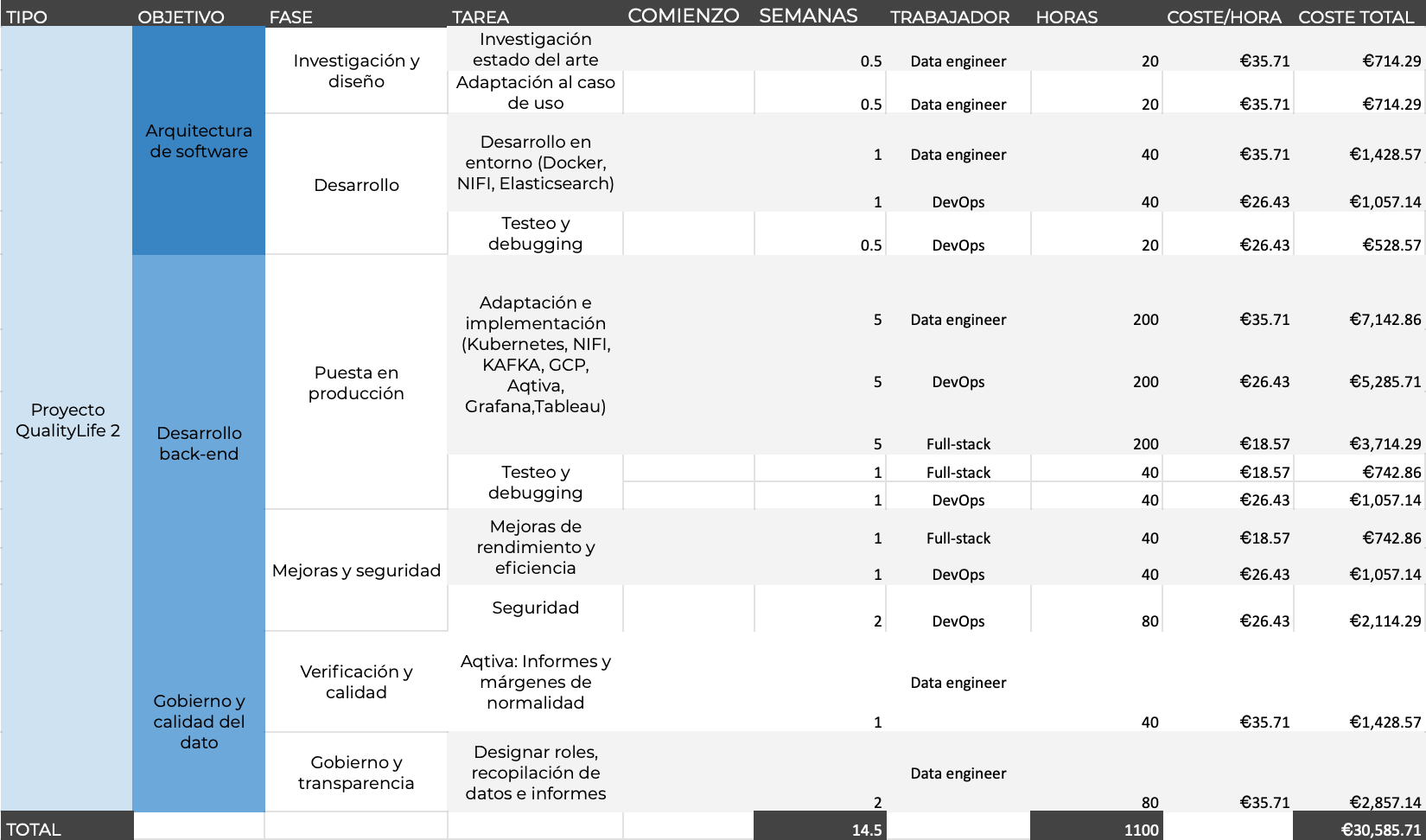
Tabla 1: Puestos y requisitos del equipo.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Puesto | Experiencia | Cantidad | Salario aprox. | URL LinkedIn |
| Data engineer | * Desarrollo de sistemas * Tecnologías distribuidas/Big Data * Certificación GCP * Data analysis * Data quality & governance | 1 | 50.000€ | <https://www.linkedin.com/in/bizquierdof/> |
| DevOps/Cloud engineer | * Certificación GCP * Experiencia en desarrollo de sistemas * Seguridad en Cloud | 1 | 37.000€ | <https://www.linkedin.com/in/moreaujulien/> |
| Full-stack developer | * Certificación AWS * SQL/NoSQL * Servicios RESTful * Experiencia en front y back | 1 | 26.000€ | <https://www.linkedin.com/in/sebastianrios/> |

5. COSTE DESARROLLO

Los costes del proyecto se pueden dividir en dos componentes, salarios e inversiones. Para determinar el coste estimado perteneciente a los salarios del equipo necesario, se ha creado un Work-Breakdown Schedule (WBS), marco comúnmente utilizado en Project Management. En este se detallan las actividades, tareas, empleados y horas imputadas en distintas etapas del proyecto para determinar tiempos y costes de una forma fehaciente.

*Tabla 2: Work-Breakdown Schedule (WBS).*



Como muestra la tabla 2, se ha dividido el proyecto en 3 objetivos, en concreto, el la investigación y desarrollo de la arquitectura (incluyendo un MVP); el desarrollo de la parte ligada al back-end, es decir, el sistema completo y; la parte orientada al gobierno y calidad del dato.

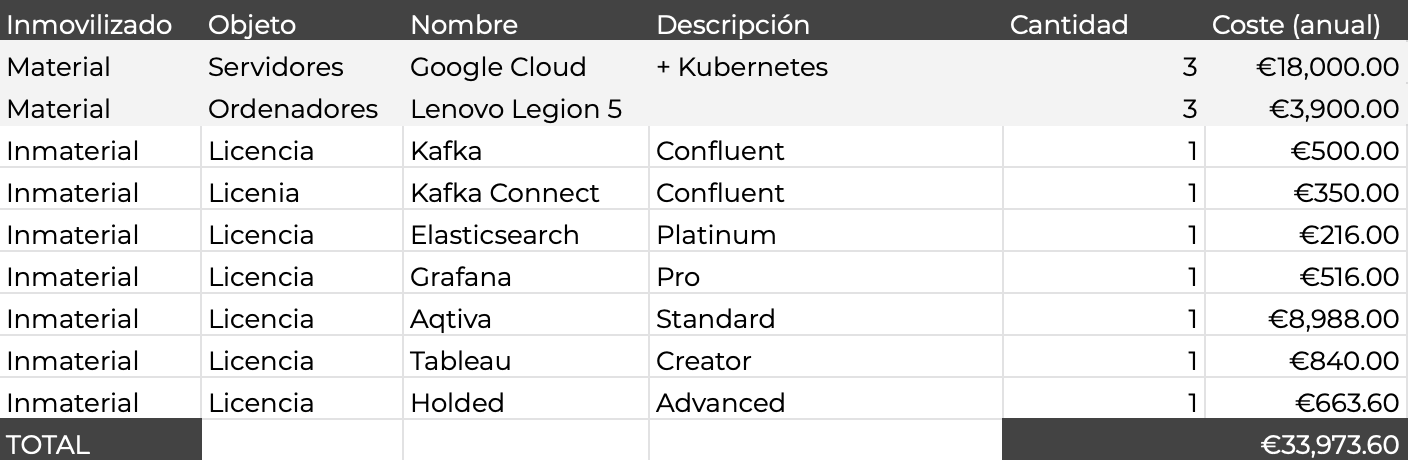
El primer módulo contará con el trabajo del data engineer y del ingeniero de DevOps/Cloud. En este, se buscará la preparación conceptual al caso de uso utilizando las tecnologías y métodos más adecuados.

El segundo módulo requerirá del desarrollador full-stack aparte de los otros dos miembros. Este será el módulo de mayor duración y técnica para desarrollar el sistema funcional al completo.

Finalmente, el data engineer será el encargado, por su experiencia senior, en determinar y elaborar la documentación y protocolos necesarios para permitir la transparencia y control de los datos de forma horizontal en la empresa.

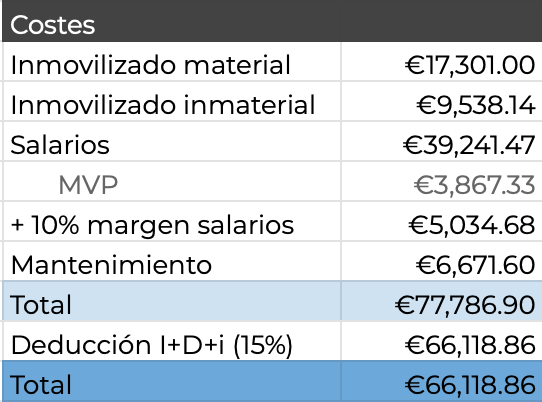
Para determinar el coste, se han estimado los tiempos necesarios por cada tarea en función de las horas laborables por semana. Asimismo, se han utilizado salarios medios en Valencia para estimar el coste por hora de cada miembro. Como se puede observar en la tabla 2, el coste asciende a 30,585.71€ sólo en cuanto a salarios. Este coste se reparte en un total de 14.5 semanas de duración total.

*Tabla 3: Costes de inversiones.*



Además de los costes salariales, se estiman los costes de inversiones en la tabla 3. Los costes materiales englobando los servidores en Google Cloud con Kubernetes y la compra de ordenadores para los 3 miembros del equipo asciende a 21.900€. Las inversiones en inmovilizado inmaterial por otro lado, las cuales abarcan la adquisición de licencias durante un año ascienden a 18,046€. Los costes derivados de las inversiones necesarias son por tanto 33,973.6€.

Tabla 4: Gastos totales para el desarrollo del proyecto.

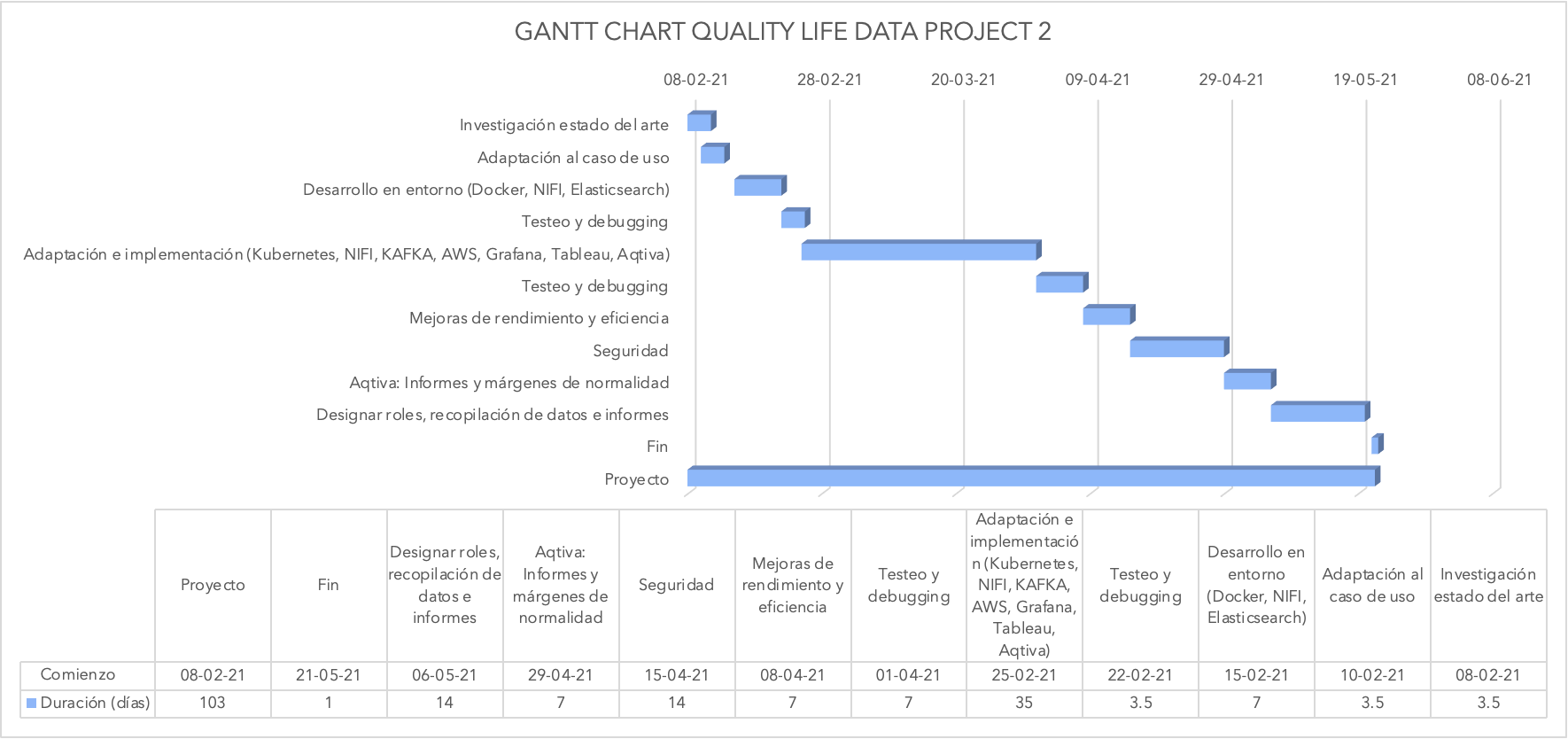


La tabla 4 muestra el desglose de gastos totales del proyecto. Asumiendo un margen del 10% de los costes salariales debido a retrasos en alguna o varias fases de desarrollo, el coste total del proyecto alcanza los 77,786.9€. Esta cifra contiene la deducción del IVA de las inversiones (i.e. -21%) así como la cotización a la seguridad social de los salarios del equipo por parte de la empresa (i.e. +28.3%). Además, se ha asumido un posible sobrecoste del 10% de los salarios debido a posibles retrasos o errores que pudieran alargar el tiempo de desarrollo del proyecto. Se ha incluido también el coste de mantenimiento por parte del desarrollador full-stack. Se ha decidido mantener a dicho trabajado en caso de que QualityLife necesitará hacer algunas mejoras ya fuese en backend (e.g. sistema o bases de datos) o en frontend (e.g. desarrollo, conexión y mejora de la web del DataProject 1). Esta persona, certificada en AWS tiene las capacidades necesarias para revisar y afrontar posibles problemas en cuanto al almacenamiento en S3 o las instancias de Google Cloud. Se asume que el coste de mantenimiento del sistema no sobrepasaría el 20% de la jornada laboral del full-stack developer. Al tratarse de un proyecto de I+D+i, Quality Life puede deducirse un 15% de los gastos al final del ejercicio, resultando así en un coste neto final de 66,118.86€.

6. TIEMPO NECESARIO PARA PUESTA EN MARCHA

Se estima que el proyecto podría realizarse en un plazo de 103 días como muestra el diagrama X. En efecto, asumiendo que el proyecto comenzase el día 8 de febrero de 2021, la fecha de finalización sería el 2 de junio de 2021 extendiéndose así en un plazo de menos de 14.5 semanas (i.e. menos de 4 meses).

*Diagram 5: Diagrama de Gantt.*

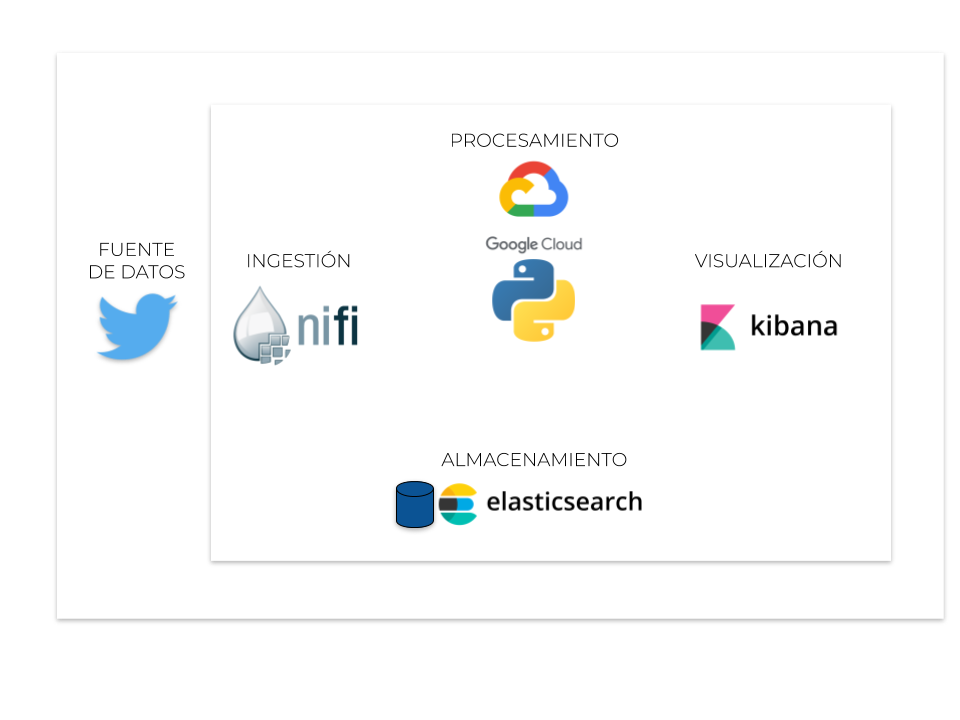


Como muestra el diagrama, las fases de mayor esfuerzo serían las de adaptación e implementación del sistema en producción y, los módulos orientados a la calidad de datos. Así, se enfatiza el esfuerzo aplicado a la transparencia y control de los datos una vez finalizado el proyecto. Se busca de este modo fomentar la horizontalidad de la información para prevenir futuros conflictos o problemas innecesarios.

7. DESARROLLO MVP

Para demostrar la viabilidad del proyecto, el Grupo Fiji ha utilizado tecnologías actuales creando un sistema de respuesta personalizada que se ajusta a las necesidades del potencial cliente casi en tiempo real de forma sencilla y asequible. El prototipo se basa en un sistema compuesto por una capa de ingestión (i.e. NIFI), una capa de almacenamiento y visualización (i.e. Elasticsearch & Kibana) y una capa de procesamiento en la Nube (i.e. Google Cloud & Python) montado dentro de un contenedor (i.e. Docker) fácilmente replicable.

*Diagrama 6: Tecnologías utilizadas en el prototipo de sistema.*



7.1 INFRAESTRUCTURA

Google Cloud es el servicio de Google donde el usuario puede alquilar computación o almacenamiento principalmente a un precio asequible. Google ofrece 300€ por cuenta gratis para uso con fines académicos. En este caso hemos hecho tal uso para abaratar costes. Google Cloud es un servicio de gran calidad, fácil de usar y a un precio competitivo en el mercado de los servicios Cloud. Es la herramienta con la que más experiencia contamos entre los proveedores Cloud. Para este prototipo hemos utilizado una máquina virtual Debian 10 con 2vCPUs, 9.75Gb de RAM y 64Gb de espacio de disco; unas capacidades suficientes para este prototipo.

Docker es la herramienta de contenerización idónea para entornos de desarrollo. Docker permite crear una máquina virtual de forma que múltiples herramientas pueden ser automáticamente ejecutadas al mismo tiempo en distintos sistemas operativos. La adecuación de Docker se evidencia en la facilidad de crear una arquitectura rápidamente en un entorno en Cloud. De este modo, la imagen 7 muestra una parte del fichero docker-compose donde configuramos las imágenes de las distintas herramientas que se utilizarán una vez levantado el contenedor.

*Imagen 7: Sección del archivo docker-compose.yml con las distintas imágenes y configuraciones.*



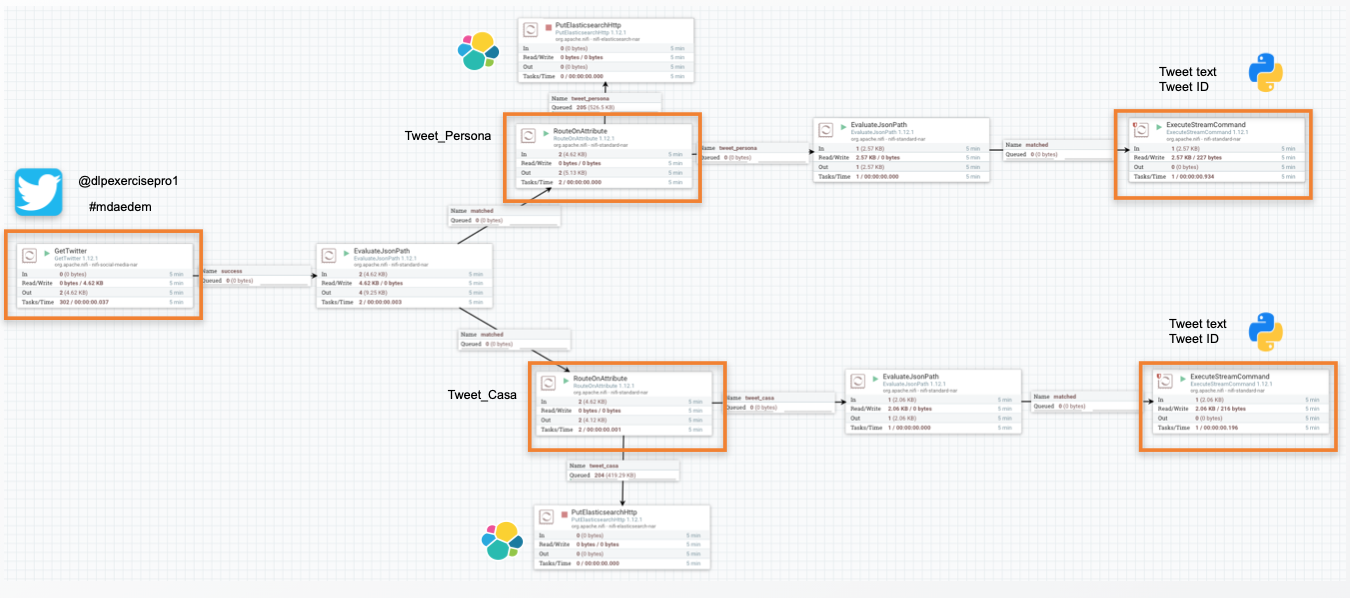
Para ejecutar Docker en el servidor en Google Cloud, hizo falta instalar Docker y docker-compose en la propia memoria del servidor.

7.2 INGESTIÓN

Para este prototipo, NIFI ha servido como herramienta de ingestión y en cierto grado de transformación (ETL). Como muestra la imagen 8, el flow diseñado ingiere los tweets de la cuenta de Twitter elegida. Tras la ingestión, bifurca el flujo según si es un tweet de oferta o de petición. En el caso de que sea de oferta, por un lado envía el tweet completo a un índice de elasticsearch, y por otro, lanza la orden de ejecutar un script en el servidor que limpia el tweet y alimenta con las coordenadas de la ciudad en cuestión. Finalmente, envía la información procesada a otro índice de elasticsearch.

Del mismo modo, en el caso de tweets de petición nifi envía los tweets sin procesar directamente a elasticsearch, a la vez que ejecuta otro script en el servidor. Dicho script se encarga de procesar la petición y cruzarla con una de las ofertas en la base de datos de elastic. Finalmente, ejecuta la función de responder al tweet y envía la unión de tweet petición y oferta de vuelta a otro índice de elasticsearch.

*Imagen 8: Flows de NIFI.*



7.3 PROCESAMIENTO

El procesamiento para este MVP se realiza mediante dos scripts de python alojados en la instancia de computación de Google Cloud. La bifurcación de NIFI en función del tipo de tweet resulta en un efecto de trigger para los scripts de procesamiento.

De un lado, cuando el tweet ingerido es de oferta, “tweet\_casa”, un procesador se encarga de lanzar la ejecución de un script, que limpia el texto para quedarse con los campos más relevantes e incluye las coordenadas de la ciudad en cuestión. Finalmente, el propio script envía los campos como un nuevo documento a un índice de elasticsearch.

De otro lado, cuando el tweet ingerido es de petición, “tweet\_persona”, otro procesador lanza la ejecución del segundo script. Este script procesa y limpia el tweet y coteja, en una primera instancia, los hobbies del usuario con una tabla ya creada que contiene las puntuaciones de cada ciudad para dichos hobbies, creadas en base a nuestro propio criterio. De este modo, y siguiendo el método de mínimos cuadrados, se obtiene un ranking de ciudades que más se aproximan a los requisitos del cliente.

Con el orden de ciudades establecido, teniendo en cuenta que el número de habitaciones deberá ser como mínimo igual o superior al número de miembros de la familia, y asumiendo que se destina el 20% del salario anual al pago del alquiler, se realiza una llamada vía API Rest al índice que contiene los tweets limpios de ofertas para encontrar la casa que mejor se adapta a los requisitos del usuario en cuestión.

Dicha query está embebida dentro de un bucle, de modo que si no encuentra una casa con las habitaciones y precio deseado en la primera ciudad del ranking, pasaría a la siguiente, y así hasta haber probado todas las ciudades.

Imagen 9: Extracto de la query a elasticsearch con los filtros deseados



Para asegurar que siempre se dé una respuesta inmediata al cliente, en caso de no encontrar la casa ideal para el usuario, la respuesta así lo indicará. Debido a que siempre se almacena la unión de casa y petición, estos casos son trazables y se puede volver a contactar con el cliente cuando se encuentre una mejor oferta.

Finalmente, una vez identificado el mejor resultado, el script utiliza el id del tweet de la petición para responder automáticamente con el resultado encontrado, dando detalles de la ciudad, precio, número de habitaciones y código.

Finalmente, como se ha mencionado, la unión establecida se envía de nuevo a elastic con el fin de monitorizar.

7.4 ALMACENAMIENTO

En particular, Elasticsearch es un sistema idóneo para documentos de texto plano, como tweets. Es una buena elección como base de datos en cuanto a rapidez y volumen. Elasticsearch ofrece la capacidad de indexar inversamente, es decir no por título sino por contenido. Para este prototipo levantamos un contenedor con una imagen de Elasticsearch en el puerto 9200.

7.5 VISUALIZACIÓN

En este prototipo, aparte de utilizar Kibana por motivos funcionales a la hora de revisar los índices y crear patrones, hemos hecho uso de varios cuadros de mando para ofrecer información comprensible y de valor añadido para Quality Life.

*Imagen 10: Dashboards de visualización de tweets.*

8. SIGUIENTES PASOS

A corto plazo, los próximos pasos van dirigidos a mejorar la experiencia de usuario, incluyendo inteligencia artificial para ofrecer una solución al usuario más humanizada y completa (i.e. ofrecer la casa que más se ajusta a las prioridades de usuario, pero buscar una segunda opción también acorde a los requisitos, pero más económica.

A largo plazo, y con el objetivo de enlazar los dos proyectos, los pasos a seguir irían destinados a explotar la información del primer proyecto para aprovecharla en este segundo. Es decir, encontrar qué ciudades son más populares, con mayor grado de éxito/satisfacción tras la mudanza de los usuarios, para crear conexiones con inmobiliarias de la zona, de las que obtener las ofertas que se publicarán a los usuarios que busquen cambiar de ciudad.