Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

Desarrollo de una aplicación web para acercar datos públicos abiertos a los ciudadanos

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Sebastiá Pérez, Daniel

Tutoras: Albert Albiol, Manuela
Torres Bosch, María Victoria

Curso 2021-2022

Resumen

El proyecto consiste en desarrollar una aplicación web en la que se muestra información pública relevante para los ciudadanos a partir de datos abiertos publicados por distintas administraciones.

La aplicación mostrará los datos de forma sencilla para permitir que el ciudadano pueda entender la información. Además, se procesarán los datos para ofrecer al ciudadano información con valor añadido, por ejemplo, con estadísticas, porcentajes, etc.

Por otro lado, se hará un diseño con opciones para personalizar ciertos aspectos de la aplicación en función de las necesidades del usuario. El desarrollo del proyecto se realizará aplicando un enfoque ágil. Las tecnologías a utilizar para el desarrollo de la aplicación serán HTML, CSS, Angular y Bootstrap entre otras.

Palabras clave: datos, datos abiertos, administración pública, portal, web, aplicación web

Abstract

The project consists of developing a web application that displays public information relevant to citizens based on open data published by different administrations.

The application will display the data in a simple way to allow the citizen to understand the information. In addition, the data will be processed to provide the citizen with added value information, e.g., statistics, percentages, etc.

On the other hand, a design will be made with options to customize certain aspects of the application according to the user's needs. The development of the project will be carried out using an agile approach. The technologies to be used for the development of the application will be HTML, CSS, Angular and Bootstrap, among others.

Keywords: data, open data, public administration, portal, web, web application

Tabla de contenidos

Índice de figuras

1. Introducción

Las administraciones públicas suelen hacer anualmente estudios y recogida de datos de diversos temas. La recolección de toda esta información se hace, como no puede ser de otra manera, con fondos públicos. Es decir, todos y cada uno de los ciudadanos ponen dinero de sus bolsillos para la obtención de estos datos. Sin embargo, el producto obtenido con esos fondos (los datos recolectados) no llega a todo el público. Siendo que en una democracia la administración pública debería servir a la ciudadanía, estos datos que se han recabado con el dinero del contribuyente deberían estar accesibles y fácilmente consumibles para todo contribuyente que quisiera acceder a ellos y utilizarlos como desee. En el caso de los datos de la Generalitat Valenciana (GVA) esos datos están disponibles, pero como se explicará en el siguiente apartado, no es suficiente.

Sin embargo, la inmensa mayoría de la población solo tiene conocimiento de estos datos a través de medios de información, como periódicos, periódicos digitales, noticias en televisión o en radio, etc. Que se hacen eco de algunos estudios llevados a cabo por las administraciones.

En este trabajo se ha intentado aportar un granito de arena para ayudar a solventar esa falla de las administraciones públicas, que no facilitan el acceso a los datos que obtienen.

1.1 Motivación

Como se comentaba en el apartado anterior, la inmensa mayoría de la población recibe los datos recabados por la administración a través de los programas, informativos y contenido que los medios decidan publicar. Y tal y como esa misma población sabe, esas cadenas suelen tener un sesgo político importante. Esto implica que a la hora de transmitir la información se hace desde la perspectiva política o ideológica de la cadena o medio en cuestión. Este trabajo no tiene nada que decir con respecto a este hecho. Sin embargo, si unos datos se recaban con fondos públicos, deberían estar fácilmente

accesibles y adecuadamente presentados para que la ciudadanía pueda consumirlos. Es muy posible que haya parte de esos datos que no sea de gran interés para la mayoría de la población (p. ej. Mapa de los tipos de grava en el suelo). Pero hay otros que sí que podría ser interesante tener de una manera accesible y fácil de entender (p.ej. Contratos laborales, bibliotecas, escolarizaciones etc.).

En el portal Dades Obertes de la GVA[1] se publican regularmente estos datos recabados por distintas administraciones públicas. Sin embargo, solo están disponibles en forma de tablas que muestran los datos en bruto. Además, la única funcionalidad que ayuda a navegar por las tablas es un buscador simple que filtra las filas de las tablas para mostrar las que contengan un término igual o similar al de la búsqueda. Esto da como resultado un portal en el que se pueden consultar y descargar los datos en bruto pero que, a no ser que el usuario esté dispuesto a procesarlos por su cuenta, podrá sacarles muy poco partido.

Como se verá más adelante y como cabría esperar, el número de consultas y descargas de estos datos es bajísimo. A excepción de los conjuntos de más actualidad como los referentes al COVID, todos los demás apenas se consultan.

Con este trabajo se intenta desarrollar una prueba de concepto de lo que un portal de datos abiertos de la administración pública debería ser. Además, será funcional, pues los datos que se presenten en la aplicación desarrollada para el trabajo serán obtenidos directamente de las APIs de la GVA de consulta de sus datos abiertos.

1.2 Objetivos

Dada la motivación del trabajo el objetivo principal de este TFG es construir una aplicación web que muestre información de interés para los ciudadanos a partir de datos abiertos ofrecidos por las administraciones públicas. Este objetivo se puede desgranar en los siguientes subobjetivos:

- Mostrar datos obtenidos directamente de los publicados en el portal de datos de la GVA.
- Que los datos se muestren de una manera cómoda, fácil de entender y útil por parte de los ciudadanos.

[1] portaldadesobertes.gva.es/

- Ofrecer un diseño responsive que se adapte a los diferentes tamaños de pantalla.
- Ofrecer datos de valor añadido, como medias y estadísticas, obtenidas mediante el procesamiento de los datos.
- Hacer cómodo el acceso a datos que se quieran consultar regularmente.

Para lograr estos objetivos, en este trabajo se presenta "Dades Útils", una propuesta de portal de datos abiertos que se ha diseñado e implementado poniendo el foco en las mencionadas carencias existentes en el portal de la GVA.

1.3 Impacto esperado

Debido a la inabarcable cantidad de datos abiertos disponibles, no se espera solventar el problema por completo, pero sí mostrar un ejemplo de cómo debería ser la solución.

Se espera que los datos elegidos puedan ser de utilidad para el usuario. Algunos ejemplos de esta utilidad pueden ser los siguiente. Si se presentaran datos de contrataciones, puede aprovecharlos alguien que, por ejemplo, quiere orientar su búsqueda de trabajo y centrarse en pocas localidades. Si se presentan datos de escolarización, podrían servir a unos padres que están planteándose si matricular a su hijo en un centro público o en uno concertado. Si se presentan datos de bibliotecas, podrían servir a estudiantes que necesiten espacios de estudio o a quien esté buscando ejemplares de libros u otros recursos que se puedan encontrar en bibliotecas.

Este trabajo va en la línea de los Objetivos de desarrollo Sostenible (ODS) creados en 2015 por la Asamblea General de las Naciones Unidas (ONU) [1]. Concretamente con el objetivo 16, Promover sociedades justas, pacíficas e inclusivas. Y más concretamente con las siguientes metas:

- Meta 16.6 - Crear a todos los niveles instituciones eficaces y transparentes que rindan cuentas.
- Meta 16.7 - Garantizar la adopción en todos los niveles de decisiones inclusivas, participativas y representativas que respondan a las necesidades.
- Meta 16.10 - Garantizar el acceso público a la información y proteger las libertades fundamentales, de conformidad con las leyes nacionales y los acuerdos internacionales.

Además, la GVA aprobó en el pasado marzo de 2021 la adhesión a la Carta internacional de datos abiertos [2][3]. Esta carta cuenta con 6 principios, de los cuales este trabajo está en relación con los 2 siguientes:

- Principio 2 - Oportunos y Exhaustivos
- Principio 3 - Accesibles y Utilizables

En esa misma línea, la GVA aprobó el 26 de marzo de 2019 el Plan Bienal de Transparencia 2019-2021 de la GVA [4]. Es un plan de aplicación entre los años 2019 y 2021 en el que se pretende llevar a cabo las propuestas planteadas durante un proceso

participativo que acabó el 4 de marzo de ese mismo año. En el plan se plantean diferentes fases para cumplir sus objetivos y 6 líneas estratégicas. Cada línea estratégica cuenta con varios puntos hasta sumar un total de 86. Los principales puntos con los que este trabajo guarda relación son los siguientes:

- Dentro de la línea estratégica 1 – Garantizar el acceso efectivo a la información pública
    - Punto 32. Mejora del diseño y visualización de los contenidos del portal GVA Oberta
- Dentro de la línea estratégica 2 – Promover la cultura de la transparencia entre la ciudadanía
    - Punto 34. Estrategia de comunicación y difusión del gobierno abierto
- Dentro de la línea estratégica 4 – Crear un marco de planificación, evaluación y rendición de cuentas de las políticas públicas
    - Punto 57. Mejora de la rendición de cuentas de la Generalitat

Cabe mencionar también el IV Plan de Gobierno abierto de España 2020-2024 [5]. España es país miembro de la asociación para el gobierno abierto desde 2011. Asociación de 78 países de todo el mundo y organizaciones que defienden la transparencia, la rendición de cuentas, la participación, la colaboración y la integridad en las actuaciones de las administraciones públicas. Desde entonces, España ha elaborado 4 planes de acción para implantar e impulsar los principios del gobierno abierto. El IV plan se estructura en 4 grandes objetivos. Este trabajo guarda relación con el segundo objetivo:

- El plan de mejora de la transparencia y rendición de cuentas a través de la mejora del portal de la transparencia y la apertura de datos y reutilización de información del sector público.

El detalle de la relación los principios de la Carta Internacional de Datos abiertos y los puntos del Plan Bienal de Transparencia 2019-2021 se puede encontrar en el Anexo I. El grado de relación con los ODS se encuentra en el Anexo II.

1.4 Metodología para el desarrollo del trabajo

Esta sección se dividirá en las practicas agiles que se van a aplicar, las fases por las que pasará cada ítem del backlog desde que se especifica hasta que se prueba con usuarios y la metodología que se aplicará con el sistema de control de versiones. En apartados posteriores se explicará cómo concretamente se han aplicado tanto las prácticas agiles como las fases del ciclo de vida de los ítems del backlog.

1.4.1 Metodología ágil

En este proyecto se va a seguir un enfoque ágil de desarrollo. Existe una gran cantidad de conjuntos de datos abiertos que hace que no sea viable abordarlos todos en este trabajo, ya que hay unos plazos marcados por la universidad para la realización de TFGs. Por ello, siguiendo un enfoque ágil se podrá aportar valor al usuario desde la primera entrega. Este enfoque disminuye el riesgo de que no se llegue a realizar alguna

de las últimas fases del proyecto, o que se detecten errores al final y no se disponga de los medios o el tiempo para solventarlos. Si se eligiera un desarrollo tradicional, y se dieran estos problemas, no se llegaría a desplegar la aplicación y los objetivos del proyecto no se cumplirían.

Por este motivo se ha decidido darle un enfoque ágil al desarrollo del proyecto. Para hacerlo se escogerán varias prácticas del catálogo de prácticas ágiles Agilev Roadmap [6]. Cuáles se han escogido concretamente y cómo se han llevado a cabo se detalla en el apartado 5.1.1 Practicas de metodología ágil.

1.4.2 Fases de cada ítem

Cada ítem del backlog que se refiera al desarrollo de alguna característica pasará por las siguientes etapas:

1. **Especificación de requisitos**. La especificación de requisitos se realizará utilizando pruebas de aceptación. Se definirán pruebas de aceptación para las tareas del backlog que más valor aporten a la aplicación o que más contacto tengan con el usuario. La definición de las que no tengan esa importancia se dejará en una breve descripción. Además los ítems del backlog que no sean desarrollos sino tareas a realizar para llevar a cabo el proyecto también se definirán con una descripción breve.
2. **Prototipado**. Para los ítems que cuenten con la implementación de un interfaz, primero se realizará un prototipo no funcional.
3. **Implementación**. Se llevará a cabo la creación del código o se realizará el trabajo necesario para construir lo requerido por el ítem. En esta fase se integrará también en el entorno de desarrollo.
4. **Pruebas de sistema e integración**. Lo siguiente será el testing y si se supera satisfactoriamente se pasarán las pruebas de aceptación. Después se integrará en el entorno de producción, donde se volverá a hacer el testing para asegurar la correcta integración.

Además, a lo largo del proyecto se realizarán 2 o 3 pruebas con usuarios en las que se evaluará la eficacia y la usabilidad de la aplicación. En estas pruebas se utilizará una medida de la satisfacción percibida por el usuario utilizando la escala SUS (System Usability Scale) [8].

1.4.3 GitFlow

Git Flow es una metodología de trabajo que define la forma de gestionar las ramas del repositorio Git (software de control de versiones). Fue definida por Vincent Driessen en 2010 y, desde entonces, se ha convertido en una de las formas de trabajo más populares y extendidas hoy en día. Git Flow define 5 tipos de ramas distintos en su modelo, 2 principales y 3 secundarias. En este proyecto se utilizará esta metodología a excepción de la rama secundaria “Release”.

1.5 Estructura de la memoria

A continuación se hace una descripción básica de las próximas secciones del trabajo.

2. **Situación actual**: En esta sección se analiza la situación de los datos abiertos a nivel europeo, nacional y autonómico. También se hace una crítica a dicha situación.
3. **Análisis y modelado**: En esta sección se presenta el modelado conceptual de los datos abiertos en la GVA y se analiza el marco legal relevante en el proyecto. También se analizan varias posibles soluciones que se podrían llevar a cabo para alcanzar los objetivos y se explica por qué se ha elegido la que se va a desarrollar.
4. **Diseño de la solución**: En esta sección se presentan varios diagramas para exponer la arquitectura del sistema. También se listan las principales tecnologías que se emplearán junto con una breve definición de cada una.
5. **Desarrollo de la solución propuesta**: En esta sección se explica cómo ha funcionado el desarrollo del proyecto, cómo se han podido aplicar las prácticas listadas en el apartado de metodología y los principales problemas y dificultades que han surgido a lo largo del proyecto. También se explica como se añadiría un nuevo conjunto de datos al portal
6. **Resultados, conclusiones y relación con los estudios**: Se exponen los resultados y se analiza si se han cumplido los objetivos iniciales. También se incluye una reflexión final sobre el proyecto y se pone en valor la aplicación y el aporte de todo lo aprendido a lo largo de las asignaturas de la carrera. También se listan y justifican las competencias transversales que más se han trabajado en el proyecto.
7. **Trabajos futuros**: En esta sección se hace una breve lista de los desarrollos que se podrían seguir llevando a cabo en Dades Útils.
8. **Referencias**: En esta sección está el listado de todas las referencias empleadas a lo largo del trabajo.
9. **Anexo I**: En esta sección se presenta el detalle de la relación del proyecto: En este apartado se explica porque el proyecto está relacionado con los diferentes documentos e iniciativas planteadas en la sección 2.
10. **Anexo II**: En esta sección se presenta el detalle de la relación del proyecto con los ODS.
11. **Glosario de términos y acrónimos**: En esta sección se definen algunos términos específicos en el contexto del trabajo.

2. Situación actual

En este capítulo se describe la situación actual relacionada con datos abiertos y aplicaciones que hacen uso de estos datos. El capítulo se ha estructurado en 5 subsecciones que abordan el tema desde entornos y puntos de vista diferentes: los datos abiertos en la Unión Europea y España, los portales autonómicos de datos abiertos, el uso del portal de datos de la GVA por parte de la ciudadanía, el derecho de

acceso a la información en España y por último otros trabajos de la ETSInf de temática relacionada.

2.1 Los datos abiertos en la Unión Europea y España

El Portal Europeo de Datos publica anualmente un informe [9] a raíz de un estudio sobre la madurez de los datos abiertos de los diferentes países miembros de la Unión Europea y de la EFTA. En este informe se elaboran unas puntuaciones en forma de porcentaje para evaluar 4 dimensiones relativas a los datos abiertos en cada país. Las dimensiones son las siguientes:

- **Política de datos abiertos**: evalúa la existencia de medidas políticas para promover los datos abiertos a nivel nacional. Incluye también la presencia de estructuras de gobierno que permitan la reutilización y la creación de iniciativas basadas en los datos. También los mecanismos dedicados a promover el descubrimiento de todas las fuentes de datos disponibles a nivel nacional
- **Portal de datos abiertos**: se centra en las funciones del portal que permiten a usuarios expertos e inexpertos acceder a los datos. También mide si los gestores de estos portales utilizan analíticas para comprender mejor las necesidades de los usuarios y tratan de satisfacerlas.
- **Impacto de los datos abiertos**: se centra en las actividades orientadas a monitorizar y medir la reutilización de los datos. Además, evalúa el impacto que han tenido los datos tanto desde un actor público como uno privado y particular en áreas como la política, social, medioambiental y económica. Evalúa la cantidad y el tamaño de las aplicaciones basadas en estos datos, el apoyo del estado y el valor económico de las mismas.
- **Calidad de los datos abiertos**: evalúa la actualización de los metadatos y de los datos reales en la medida de lo posible, el control del cumplimiento de la normativa de metadatos y la calidad del despliegue de los datos publicados. Promueve la calidad de los datos en todos los sentidos: utilizar formatos de datos abiertos, legibles por maquina y adecuados a un enfoque de datos enlazados.

Para saber interpretar los resultados hay que consultar la metodología empleada para elabora el informe, que se encuentra en un documento aparte [10]. En resumidas cuentas, cada dimensión se puede desglosar en varios apartados. Los porcentajes se calculan dando un peso a cada apartado dentro de la dimensión y sumando la puntuación obtenida en cada uno. En la Figura 1 se pueden observar las puntuaciones medias ponderadas del conjunto de todos los países en cada dimensión.

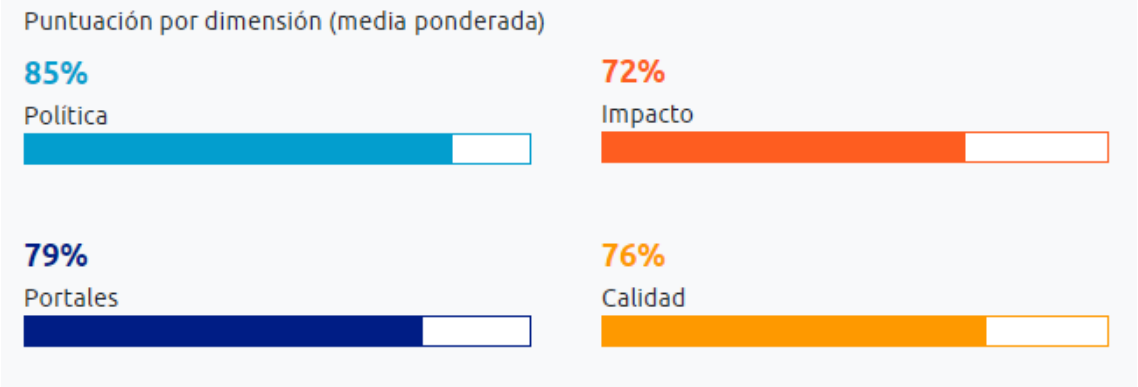

Figura 1: Puntuación media ponderada general por dimensión, estudio europeo de madurez de datos abiertos.

Además, se pueden consultar las evaluaciones concretas de España en la vista detallada por país [11]. Se puede observar que hay unos valores aparentemente buenos. Es más, en el caso concreto de España los valores son mucho mejores que la media mostrada. Se puede apreciar en la siguiente Figura 2.

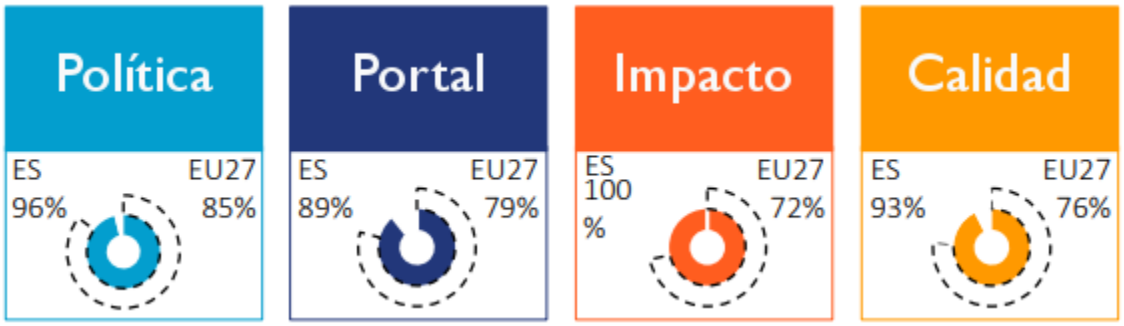

Figura 2: Puntuación ponderada, comparativa España-Europa, estudio europeo de madurez de datos abiertos.

Viendo estos datos parece que España es "Creador de tendencias" en cuestión de datos abiertos y de hecho en esa categoría es en la que el país está clasificado.

2.2 Los portales autonómicos de datos abiertos

Para obtener esa información podemos ir a el estudio realizado desde la Facultad de Comercio, Turismo y Ciencias Sociales "Jovellanos" de la Universidad de Oviedo. Titulado "Análisis multidimensional de los portales de datos abiertos autonómicos españoles" [12] evalúa diferentes aspectos de los portales autonómicos españoles.

Los aspectos más relevantes en este caso son Aplicaciones y Funcionalidades. Con respecto a las aplicaciones el estudio explica que a pesar de los datos publicados hay una muy escasa reutilización y además de eso, la mayoría de las aplicaciones reutilizadoras han sido desarrolladas por las propias comunidades autónomas. Además, solo el 5% están orientadas a la mejora de la transparencia. En el apartado de Funcionalidades, ítems como un diseño moderno y funcional, buscador con diferentes

criterios etc. obtienen una valoración positiva. Sin embargo, en el apartado estadísticas los portales que cuentan con ellas se limitan a estadísticas del uso del portal.

2.3 El uso del portal de datos de la Comunidad Valenciana por parte de la ciudadanía

En el caso de la Comunidad Valenciana se limita a presentar las estadísticas de visualización y de descarga. Basándonos en estas estadísticas es posible hacerse una idea del bajísimo uso que le dan los ciudadanos a estos datos. En la Figura 3 se presenta una estadística sacada del propio portal de datos abiertos de la GVA. Estas graficas pertenecen al informe que se publicó comparando el uso del portal en los años 2020 y

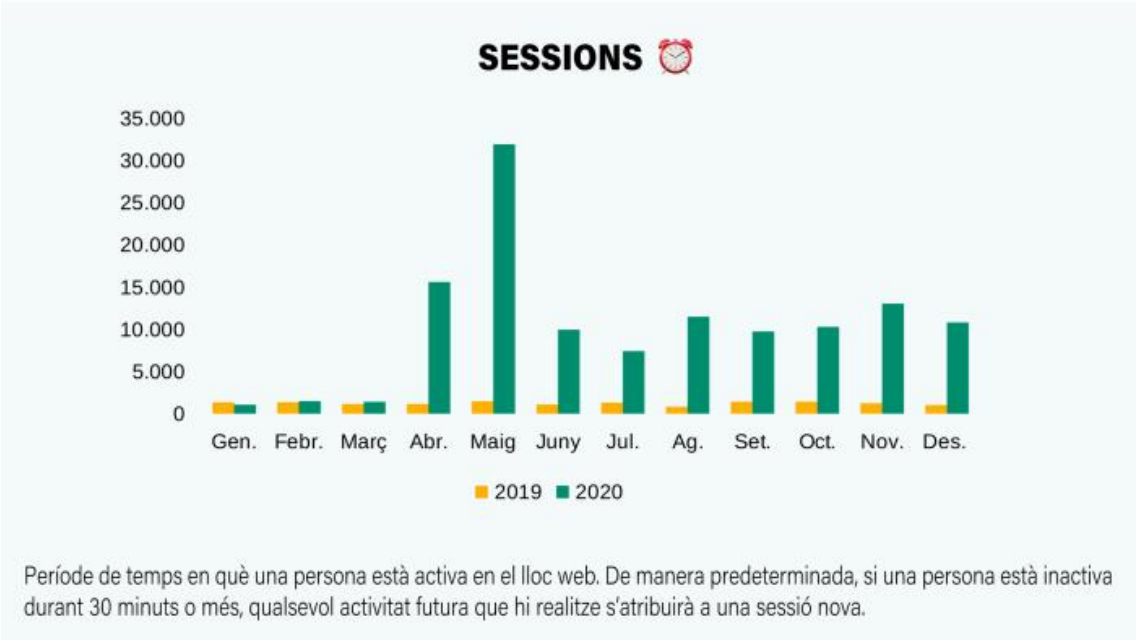

Figura 3: Gráfico del número de sesiones en el portal de datos abiertos de la GVA.

2019[13].

Se puede apreciar una diferencia abismal entre el uso que se le daba al portal en el año 2019 y el que se le dio en el 2020. En 2019 apenas tenía alguna visita mientras que a partir de abril de 2020 empezó a tener algo de uso. Se presentan la Figuras 4 y la Figura 5, otras dos gráficas relacionadas que apoyan la anterior.

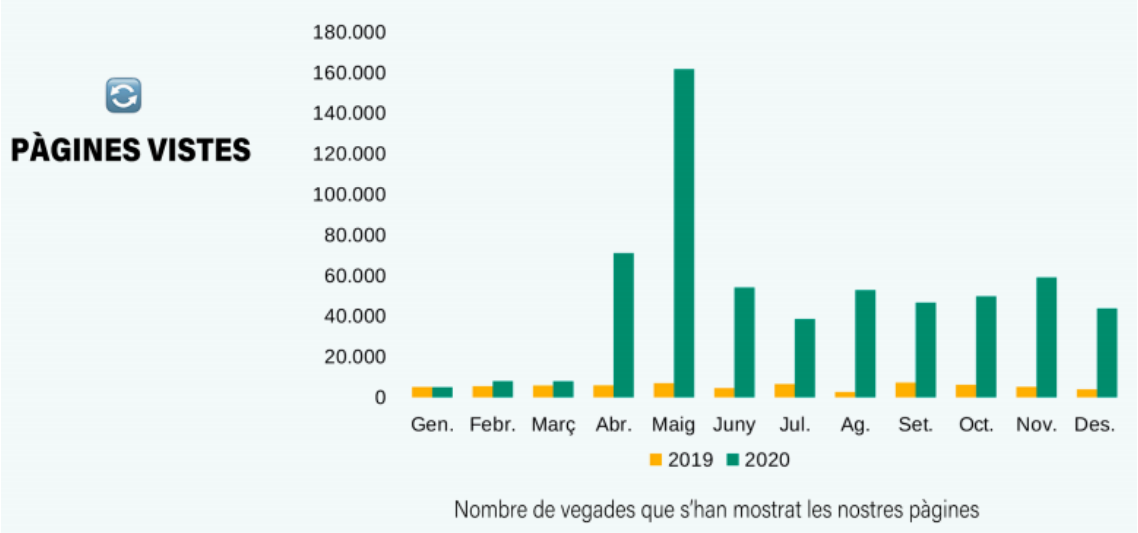

Figura 4: Gráfico de la cantidad de páginas visitadas del portal de datos abiertos de la GVA.

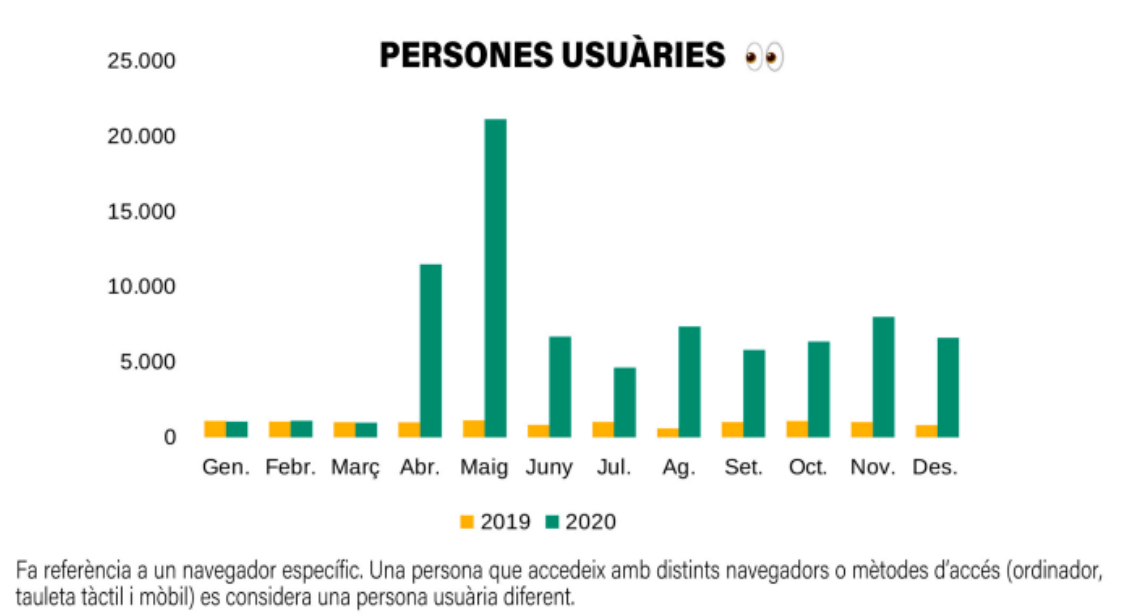

Figura 5: Gráfico del número de usuarios del portal de datos abiertos de la GVA.

Se puede observar que efectivamente en 2019 el uso del portal era casi nulo y solo a partir de abril de 2020 se empezó a consultar. Esta fecha coincide con el estado de alarma y el momento de mayor impacto social de la pandemia de COVID 19. Las fechas en que la gente estaba confinada y la población demandaba información sobre las cifras de contagios, ocupación de camas y demás datos relacionados. Además, se puede confirmar esta causalidad con la gráfica de la Figura 6, que se encuentra en el mismo informe.

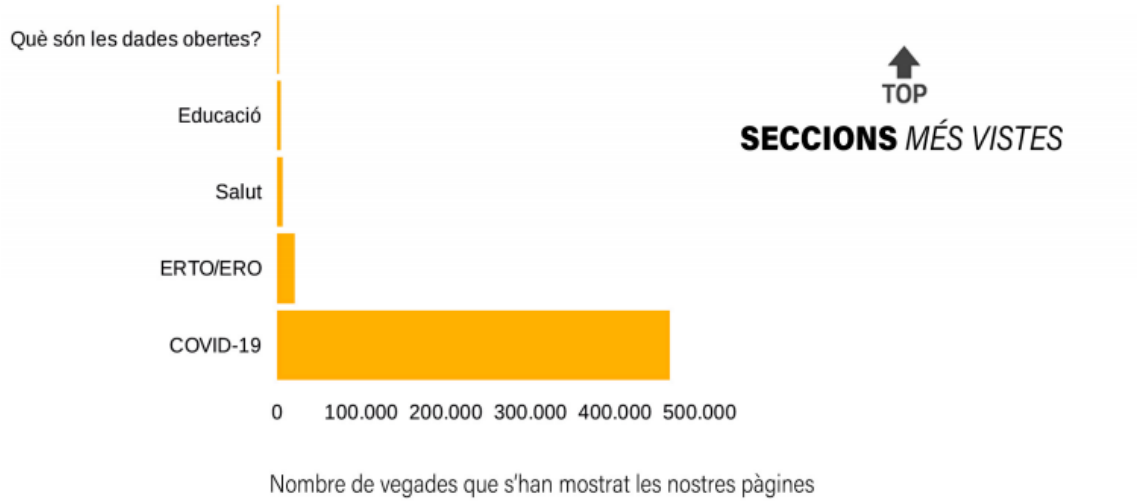

Figura 6: Gráfico de secciones más visitadas del portal de datos abiertos de la GVA.

Como se puede observar en la Figura 6, los datos relacionados con la COVID 19 son los más consultados con una grandísima diferencia. Estos datos han sido consultados más de 20 veces más que la segunda categoría más visitada.

2.4 El derecho de acceso a la información en España

Los ciudadanos podemos ejercer nuestro derecho al acceso a la información pública, acorde al artículo 105 de la constitución [14]. También está recogido en la Ley 19/2013, de 9 de diciembre, de transparencia, acceso a la información pública y buen gobierno [15]. Conviene aclarar que la "información pública" son los contenidos o documentos, cualquiera que sea su formato que hayan sido elaborados o adquiridos por las administraciones públicas. De esta manera se puede solicitar cualquier información que tenga la administración (salvo algunas excepciones), sin importar el soporte, ya sea físico o digital. La motivación de estas medidas es que tener acceso a la información pública es fundamental para fomentar la transparencia del Gobierno y de las instituciones públicas y que propicia una sociedad más justa y democrática.

Además, cuando se solicita alguna información la solicitud debe ser atendida en 1 mes como norma general y solo podrá ser denegada en algunos casos concretos como que sean perjudiciales para: seguridad nacional, defensa, relaciones exteriores o seguridad pública etc. Además, hay algunas informaciones que están sujetas a un acceso especial como procedimientos en curso o datos de carácter personal de terceros.

2.5 Otros trabajos de la ETSInf de temática relacionada

Hay varios trabajos académicos en RiuNet relacionados con los datos abiertos. Algunos de los más relevantes son los siguientes:

- "Enriquecimiento de Datos Abiertos. Desarrollo de una Aplicación para gestionar y mejorar datos abiertos basada en el Stack MEAN"[2], un trabajo de 2017 que combina los datos abiertos con otras variables externas para enriquecerlos.
- "Plataforma Serverless de Procesado de Datos Abiertos"[3]. En este TFG se crea una aplicación donde se puede visualizar información de los datos abiertos proporcionados por la ciudad de Madrid sobre las incidencias de tráfico. En la aplicación se hace una visualización de estas incidencias.
- "PoliticApp: Conoce la vida política del Congreso"[4] se crea una aplicación móvil donde se puede consultar la información de la vida política del congreso y generar opiniones al respecto.

Existen más trabajos académicos relacionados con los datos abiertos pero los tres presentados ejemplifican bien lo que se ha hecho hasta ahora y su filosofía.

Sin embargo, ninguno de ellos cumple con el objetivo presentado en este trabajo. Para empezar, "Enriquecimiento de Datos Abiertos. Desarrollo de una Aplicación para gestionar y mejorar datos abiertos basada en el Stack MEAN" utiliza información adicional además de los datos abiertos. En este trabajo se van a utilizar únicamente los datos abiertos publicados por las administraciones públicas. Pues si pretende ser una prueba de concepto del tipo de contenido que las propias administraciones deberían proporcionar, debe centrarse únicamente en la información que estas tengan disponible. Aunque cabe decir que en el trabajo a desarrollar sí que se utilizarán servicios externos para procesar los datos, como por ejemplo un sistema de geocoding para obtener las coordenadas de las direcciones publicadas.

El segundo trabajo "Plataforma Serverless de Procesado de Datos Abiertos" es un ejemplo de entre varios en los que se escoge una temática y se muestran los datos abiertos de una manera cómoda y fácil de consumir. La gran diferencia entre el trabajo propuesto y estos ejemplos es que en el propuesto no se limita a un único conjunto de datos o una única temática. El objetivo es que, aunque finalmente solo se muestren datos de un conjunto, la aplicación desarrollada responda a un esquema en el que se

[2] riunet.upv.es/handle/10251/93791

[3] riunet.upv.es/handle/10251/153189

[4] riunet.upv.es/handle/10251/88472

puedan incluir más. Puesto que es tan solo una prueba de concepto de un portal en el que se debería incluir información sobre cada conjunto de datos que se publique por parte de las administraciones.

Por último, "PoliticApp: Conoce la vida política del Congreso" sería el trabajo más similar al propuesto de los tres. Las principales diferencias son en la temática de los datos mostrados y la motivación del proyecto. PoliticApp se centra en cierta información del Congreso mientras que el trabajo propuesto aquí presenta una aplicación web diseñada para abarcar un grupo de datos creciente de varias temáticas. Además los proyectos tienen objetivos diferentes; PoliticApp propone una aplicación que intenta hacer surgir en la ciudadanía un interés por la vida política del Congreso. El trabajo aquí propuesto pretende ser un ejemplo de lo que la administración debería implementar.

2.6 Crítica a la situación actual

Como se ha podido ver en el primer punto, España está entre los mejores países de la Unión Europea con respecto a la madurez de sus datos abiertos. Sin embargo, al examinar los puntos evaluados dentro de cada dimensión, vemos que el que los datos sean fácilmente consumibles por parte de la ciudadanía no se evalúa de forma directa en el informe. La filosofía del informe solo concibe como reutilización de los datos la creación de aplicaciones y negocios basados en los datos. Aunque no lo excluye explícitamente, no incluye la consulta y el consumo de esos datos directamente por parte de la ciudadanía.

Además, centrándonos en España, la publicación de esos datos tiene poco uso más allá del que le puedan dar las propias administraciones. En el caso de los datos publicados por la GVA su uso era prácticamente nulo hasta la irrupción de la pandemia. Y aun ahora, prácticamente los únicos datos que se consultan son los relativos a las cifras de la COVID. Aunque los datos están disponibles y fácilmente reutilizables, si dejamos al margen la reutilización que hacen las propias administraciones, el impacto en la ciudadanía es prácticamente nulo.

Este trabajo parte de la filosofía de que tener los datos disponibles no es un objetivo último sino un paso básico para el que debería ser el objetivo final, que los ciudadanos pudiesen acceder a la información que proporcionan los datos recabados. En los portales de datos europeo, español, de la comunidad valenciana y de la ciudad de Valencia, los datos se proporcionan en crudo. Esto es muy adecuado en aras de la transparencia. Pero no es suficiente.

Para transmitir la filosofía de este trabajo de una manera sencilla se puede utilizar la siguiente analogía. El jefe de una empresa tiene ciertos problemas o campos en los que quiere mejorar así que contrata a un asesor que analice esos campos para encontrar una solución. Al cabo de un tiempo asesor y jefe se reúnen para decidir las medidas a aplicar. Ningún jefe con expectativas de futuro se conformaría con que el asesor le dictase cuales son las mejores medidas, sin justificarlas, para que las aplique ciegamente. Y en caso de que el jefe le exigiese la explicación o el razonamiento detrás

de esas medidas, tampoco se conformaría con que el asesor le diese su ordenador, con el que ha estado trabajando, para que el jefe se apañase buscando la información y procesándola para obtener conocimiento de ella.

Con respecto a los datos abiertos nos encontramos en esa segunda situación. Las administraciones toman decisiones, llevan a cabo reformas y gestionan los recursos que toda la población les entrega. Y esta población, para la que las administraciones trabajan y a la que deben rendir cuentas tienen que confiar en que estas decisiones tienen un buen criterio razonado o fiarse del apartado de introducción o motivación de cada propuesta. Este trabajo defiende que eso no debería ser así. Al igual que el asesor del ejemplo debería presentarle al jefe los datos en los que se ha basado de una forma en que este los pueda entender, los datos abiertos de los organismos públicos deberían presentarse no solo en bruto sino también procesados para que la ciudadanía los pueda comprender.

Además, esto es solo la punta del iceberg. Tener unos datos presentados de una manera fácil de consumir repercutiría en facilitar y con ello mejorar la labor periodística. Aún más, permitiría a cada ciudadano no depender de medios o empresas reutilizadoras para consumir esos datos y darles utilidad en su vida. A continuación, se presentan unos pocos ejemplos de casos en que el acceso fácil a esta información tendría valor para los ciudadanos.

- Un ciudadano que está buscando empleo puede estar interesado en conocer cuáles son los municipios donde se están registrando más contratos de trabajo para su franja de edad o su sector.
- Un hijo que va a aprender un nuevo instrumento y sus padres, que no tienen conocimiento al respecto, tienen que apuntarle en algún conservatorio. Podrían mirar los datos de donde hay más alumnos matriculados para ese mismo instrumento como indicador de la calidad de la enseñanza.
- Un estudiante adulto quiere estudiar en una biblioteca, pero no sabe cuál es la que más cerca le queda.
- Un investigador busca un ejemplar de un libro descatalogado muy difícil de encontrar. En vez de buscar el contacto de cada biblioteca municipal por individual podría acceder a la aplicación para obtener la información de todas ellas.
- Para promover la transparencia y que los ciudadanos tuvieran herramientas para emitir un voto informado podrían ojear los gastos de la caja fija en vez de tener que depender de empresas privadas o públicas de la información.

Estos son solo algunos de los ejemplos que podrían darse. Sin embargo, la información recabada y publicada en crudo es mucha; partiendo de cosas de un interés obvio para los ciudadanos como las cifras de contagios de COVID y la evolución de la pandemia hasta temas más especializados como los tipos de grava del suelo del territorio valenciano. Y aunque a priori algunos de estos datos puedan parecer que carecen de valor; se han recabado pagados por y en nombre de los ciudadanos así que deberían proporcionarse adecuadamente. Según hemos visto, el consenso acerca de lo correcto

que es el compartir los datos abiertos es general pero que las administraciones aporten valor a los ciudadanos con esos datos directamente y sin depender de terceros que los reutilicen no parece ser algo tan básico.

2.7 Propuesta

Este trabajo presenta una prueba de concepto del tipo de portal que se debería implementar por parte de las administraciones autonómicas o por parte de cualquier administración que tenga datos abiertos que compartir. Con dicho objeto, la propuesta debería cumplir 3 puntos fundamentales sin los cuales no se lograrían todos los objetivos anteriormente planteados:

- Presentar la información a partir de los datos procesados. Para que los ciudadanos puedan obtener conocimiento útil. Este procesamiento se debe llevar a cabo con cierto conocimiento de la temática de los datos en cuestión pues solo de esta forma se aportará valor real a los ciudadanos que se quieran beneficiar de él. Debido a este conocimiento requerido, en este trabajo no se va a abarcar una gran cantidad de temáticas ni conjuntos de datos.
- Enlazar a la información en crudo. Proporcionar un enlace a la fuente de la información mostrada para favorecer la transparencia y la rendición de cuentas.
- Enlazar a las guías para su reutilización por máquina y APIs. Estas guías las genera la GVA y son las que empleará el proyecto para obtener los datos abiertos.

  Esta medida proporciona un punto de anclaje tremendamente útil para todo tercero que quiera reutilizar estos datos para prestar cualquier servicio por medio de una aplicación basada en ellos e incluso generar riqueza.

En el trabajo no se hará especial énfasis en el enlazado a los datos en bruto o a las guías de las APIs, puesto que el portal de la GVA ya cumple esto correctamente. Se centrará en el primer punto, pues es lo que al portal mencionado y a todos los demás les falta.

3. Análisis y modelado

En este capítulo se presenta un análisis y modelado del problema abordado. En primer lugar, se construye el modelo conceptual del problema abordado en el TFG. En segundo lugar, se realiza un análisis legal referente a la reutilización de los datos y al almacenamiento de direcciones de correo. Por último, se consideran otras soluciones que podrían cumplir con el objetivo de este trabajo.

3.1 Modelado conceptual

Para conocer los conceptos del dominio del proyecto y tener una visión unificada del mismo, la figura 7 muestra un modelo conceptual del problema abordado en este TFG sobre el tratamiento de los datos abiertos.

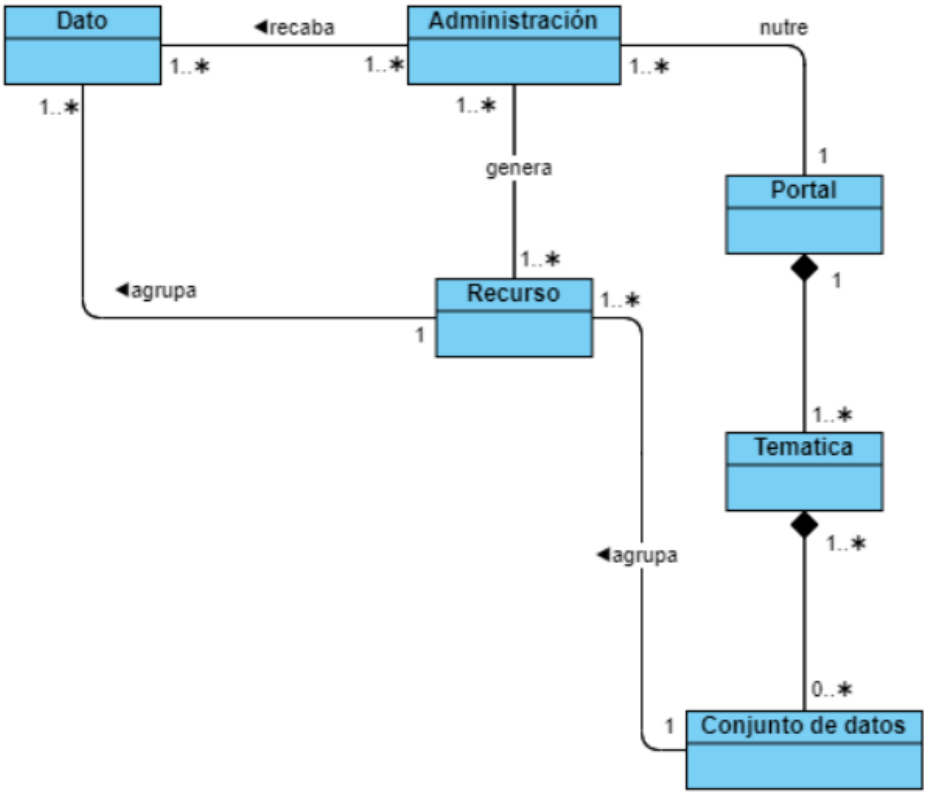

Figura 7: Modelo conceptual del tratamiento de datos abiertos.

Para empezar, son las diferentes administraciones las que recaban los datos en bruto. Para ello pueden emplear variedad de métodos. Pueden ser encuestas individuales a los ciudadanos, sensores de todo tipo, como de lluvia, polen o contaminación; estudios etc. Todos estos datos se agrupan en un archivo o tabla de base de datos, es lo que se llama un recurso. Un recurso podría ser, por ejemplo, el número de contrataciones en cada municipio en el mes de enero de 2022. Estos recursos también se agrupan en conjuntos de datos. Un conjunto de datos podría ser, por ejemplo, el número de contrataciones por municipio en todo 2022. Dentro del ejemplo, este conjunto de datos debería tener un recurso por cada mes. Es decir, habría un recurso con las contrataciones de enero, otro con las de febrero, otro con las de marzo etc. Sin embargo, cabe mencionar que la GVA no siempre es tan regular en la publicación de datos. Hay veces en las que crea un conjunto de datos para cada recurso, pasa de subir un recurso mensual a uno anual u otras irregularidades. Esto ha sido uno de los mayores obstáculos en este proyecto. Siguiendo con el tratamiento de los datos, los conjuntos de datos se publican en el portal que tiene la GVA con este propósito, donde se agrupan por temáticas como pueden ser: salud, medio ambiente, empleo, cultura etc. En el portal se puede encontrar información sobre cómo hacer uso de las APIs y se pueden visualizar los datos en bruto entre otras cosas.

3.2 Marco legal

Hay dos partes del proyecto sobre las que conviene aclarar el marco legal.

3.2.1 Uso de los datos abiertos

En cada conjunto de datos del portal hay un aviso de que está bajo una licencia Creative Commons Attribution OpenData. Si vamos a la explicación de esta licencia [16] establece dos categorías de medidas con respecto a la regulación de la reutilización del contenido. En primer lugar los "Permisos requeridos", son aquellos que necesariamente se tienen que cumplir. Entre estos se encuentran: uso, redistribución, modificación, separación etc. Después están las "Condiciones aceptables", son requisitos o medidas que el propietario de los datos puede o no aplicar a voluntad. Entre estos se encuentran: atribución, integridad, aviso, fuente etc.

De esta manera ya se conoce gran parte del marco legal que regula la reutilización de los datos. Sin embargo, para saber cuáles de las Condiciones aceptables se aplican, se debe acceder a la sección" Aviso legal" del portal de datos. En el apartado "Reutilización de la información pública" informa de que no hay licencias previas que apliquen a la reutilización de los datos más que las condiciones generales detalladas en el artículo 63 del Decreto 105/2017, de 28 de julio, del Consell.

Se listan a continuación las condiciones definidas en el citado artículo que afectan a este proyecto:

1. La reutilización no puede alterar el contenido ni desnaturalizarlo
2. Se debe citar la fuente de la información reutilizada
3. Se debe incluir una mención expresa a la fecha de actualización de la información reutilizada. Siempre que esté incluida en la información original.
4. No se puede indicar ni sugerir que las administraciones apoyan la reutilización que se ha hecho ni el producto en el que se realiza.
5. Cuando la información contenga datos de carácter personal, la reutilización deberá efectuarse siempre de forma disociada.

Con esto concluye el análisis del marco legal con respecto a la reutilización de los datos.

3.2.2 Almacenamiento de direcciones de correo

Como se verá más adelante en los ítems del backlog y en la especificación de sus requisitos, una de las funcionalidades de la aplicación a desarrollar implica el almacenamiento de direcciones de correo electrónico. Por lo tanto, conviene analizar el marco legal que envuelve a esta medida.

El Reglamento General de Protección de Datos (RGPD) y la Ley Orgánica de Protección de Datos y Garantía de los Derechos Digitales (LOPDGDD) son las dos normativas que regulan el tratamiento de datos que se va a realizar en esta aplicación.

Afectan directamente a los requisitos para manejar datos de usuarios y enviarles boletines informativos. Teniendo en cuenta estas normativas, la aplicación deberá implementar una página de política de privacidad en dos capas informativas. Además, al dar los datos el usuario tendrá que dar su consentimiento explícito. Sin este consentimiento no se pueden almacenar sus datos. Un detalle que puede parecer menor

pero que conviene no pasar por alto es que las casillas para dar el consentimiento no pueden estar marcadas por defecto.

Finaliza aquí el análisis del marco legal del proyecto.

3.3 Identificación y análisis de las soluciones posibles

Tal y como se ha desarrollado en la introducción, el problema a resolver es, muy resumidamente, que los ciudadanos no tienen acceso a la información que se puede generar a raíz de los datos recabados por las administraciones. Los fondos para recabar esos datos son públicos y aun así la administración, representativa de esos ciudadanos no pone las herramientas ni los medios para que estos puedan hacer un uso directo de los datos. Al menos sin tener que llegar al extremo de procesarlos por ellos mismos con los prohibitivos conocimientos técnicos que esto requiere.

Posibles soluciones para este problema podrían ser las que se explican a continuación.

3.3.1 Soluciones alternativas

Solicitar una nueva funcionalidad al portal

El portal incluye un formulario de contacto. Uno de los campos a rellenar en el formulario es el motivo de contacto. Nos permite elegir de entre varias opciones de un desplegable y una de ellas es “Solicito una nueva funcionalidad del portal”. La solución sería entonces, enviar una de estas solicitudes pidiendo que añadan una vista personalizada para cada conjunto de datos que cumpla con los primeros objetivos del proyecto: que se muestren datos de valor añadido calculados a partir de los datos en bruto y que la información se presente de una manera cómoda y fácil de consumir.

Para analizar la viabilidad de esta solución se han enviado tres consultas mediante el formulario de contacto. Pasados dos meses aún no se ha obtenido respuesta más allá del acuse de recibo que se envía al instante y automáticamente.

Esto no es una prueba irrefutable de que esta solución no es viable. Pero teniendo en cuenta la poca atención que parece que se pone a las solicitudes y añadiendo que la funcionalidad que se pediría es muy ambiciosa, cabe esperar que la implementación de lo pedido deje bastante que desear, ya sea en plazos o en funcionalidad.

Crear un consultor avanzado de los recursos del portal

Siendo que las APIs de consulta del portal de datos permiten realizar consultas SQL se podría implementar una interfaz de usuario en una aplicación web para realizar estas consultas. De esta manera los usuarios podrían hacer consultas complejas para obtener la información procesada a partir de los datos. Incluso se podría desarrollar algún sistema para hacer la creación de las consultas un proceso más fácil.

El análisis de esta solución ha concluido en que, si bien sí que tiene potencial para solventar el problema, es muy probable que la solución alcanzada no fuese lo suficientemente intuitiva como para que el público en general fuese capaz de utilizarla a

aun alto nivel. Además, es muy probable que muchos usuarios estén interesados en la misma información de cierto conjunto de datos. Partiendo de esta premisa, que cada usuario tenga que aprender a generar la consulta para obtener esa información y que tenga que crearla, importarla o buscarla cada vez que quiera consultarla de nuevo conlleva un gran desaprovechamiento de las ventajas de la digitalización y las posibilidades de automatización.

3.3.2 Solución propuesta

Por último, la solución propuesta es una prueba de concepto de cómo debería ser la funcionalidad que implementase el portal. Es decir, para cada conjunto de datos se creará una página donde se muestren los datos del conjunto de una forma fácil de entender, utilizar y cómoda de visualizar; con gráficas y estadísticas. Además, se enlazará a la previsualización en bruto de los datos proporcionada ya por el portal. También se presentarán procesamientos básicos de los datos que faciliten el extraer información y conocimiento de ellos. Con esto se cumple lo básico de la solución, pero además se completará con funcionalidades como la posibilidad de una suscripción a una lista de difusión por correo, funcionalidades que quedarán especificadas en los requisitos de cada ítem del backlog.

Esta solución tiene un mayor coste en esfuerzo que las descartadas en el apartado anterior, pero es sin duda la que mejor cumple los objetivos planteados. Gran parte del esfuerzo extra con respecto a las demás soluciones viene de tener que hacer una representación personalizada para cada conjunto de datos. Para esto lo ideal sería contar con conocimiento experto de la temática de cada conjunto, pero como no se dispone de él, se escogerán algún conjunto de datos más accesible por parte del desarrollador, como los datos sobre bibliotecas o contrataciones. Esta es la mejor manera de que la prueba de concepto sea lo más fiel posible a lo que debería ser la solución final al problema.

4. Diseño de la solución

Se detalla a continuación el diseño de la solución a desarrollar dividiéndolo en 3 apartados: arquitectura del sistema, diseño detallado, y tecnologías a utilizar.

4.1 Arquitectura del sistema

Como todas las aplicaciones web, la arquitectura desde un punto de vista de más alto nivel será de cliente-servidor.

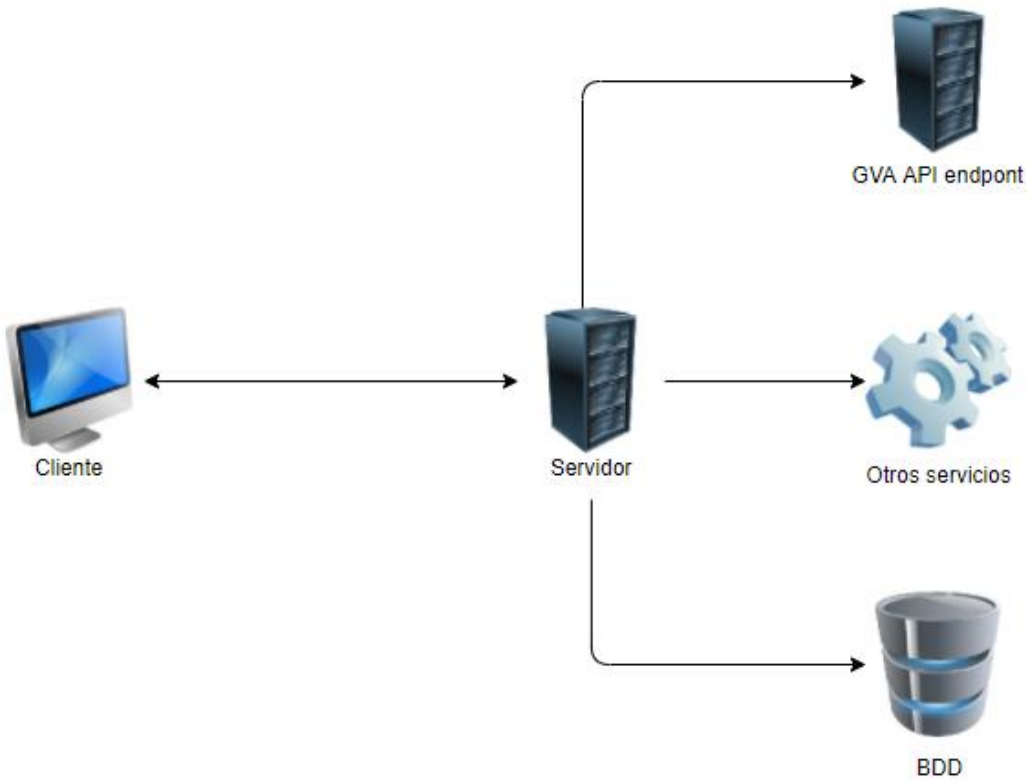

Figura 8: Modelo de la arquitectura general de la aplicación.

La Figura 8 muestra la arquitectura aplicada en la aplicación desarrollada. En esta aplicación el lado del cliente mostrará al usuario los datos obtenidos de las APIs de datos abiertos de la GVA. Como es habitual, será el lado del servidor el que se comunique con la base de datos y con los servicios de terceros. Las respuestas de las APIs de la GVA no tienen configuradas las cabeceras CORS así que se hace necesario que sea el servidor el que se comunique con ellas, actuando así también como un proxi del cliente. De esta manera evitará que los navegadores bloqueen las respuestas con los datos por no cumplir con la política CORS.

Yendo a la arquitectura a más bajo nivel podemos dividirla en dos partes muy diferenciadas; la arquitectura del lado del servidor y la del lado del cliente. Se explica cada una a continuación.

4.1.1 Arquitectura del lado del servidor

Para el lado del servidor se va a utilizar una arquitectura en gran parte monolítica. Esto es que el desarrollo del lado servidor se realizará como una sola unidad de software. Tendrá cierta estructura interna, pero al momento de compilarse se empaqueta como una solo pieza, de tal forma que todos los módulos y librerías se empaquetarán junto con la aplicación principal.

De esta manera el desarrollo es mucho más asequible en términos de esfuerzo. Para la solución final, y dependiendo de los recursos que pueda invertir la administración,

podría no ser la arquitectura ideal puesto que debería escalar bastante junto con los conjuntos de datos. Sin embargo, para una prueba de concepto cumple muy bien con las necesidades. Aporta además la facilidad para corregir fallos, hacer despliegues y pruebas.

Esto se completará con un despliegue en la plataforma Heroku. La cual brindará algunas características de la arquitectura serverless como el mantenimiento del hardware y la disponibilidad de la aplicación. Se integrará con Git para simplificar los despliegues.

También tendrá unas mínimas características de una arquitectura de microservicios ya que no será totalmente independiente, como es el enfoque monolítico puro. Tiene que actuar como proxi del cliente frente a la API de datos de la GVA.

Por último cabe decir que el servidor funcionará siguiendo una arquitectura Modelo Vista Controlador (MVC).

4.1.2 Arquitectura del lado del cliente

Para el lado del cliente se va a desarrollar una arquitectura Single Page Application (SPA). Esto es que el contenido se maneja todo desde una única página. De esta manera conseguimos una navegación más fluida y sobre todo una menor carga para el servidor, que no tiene que estar devolviendo paginas continuamente. Esto es muy conveniente puesto que el servidor va a estar desplegado en una plataforma gratuita con unos medios limitados.

En principio una Single Page Application implicaría que al entrar en la página, el cliente tendría que cargar todo el código. Sin embargo, esto se puede evitar con un framework que permita la carga perezosa como Angular, que es el que se va a utilizar. Aunque parezca contraintuitivo, con las aplicaciones Single Page Application no se pierde la navegación por URLs y las funciones del navegador que la acompañan. Las URLs se siguen utilizando como parámetros para cargar y mostrar uno u otro contenido.

Como se ha comentado en apartados anteriores, el lado del cliente tendrá que servirse del servidor como proxi para las consultas a las APIs de datos de la GVA. Con la arquitectura escogida tenemos la ventaja de que podemos cargar los datos dinámicamente.

Una de las principales desventajas de esta arquitectura es el SEO, que, aunque hay herramientas para minimizar el daño, se ve perjudicado. Sin embargo, en el caso de este proyecto esto no es mayor inconveniente, puesto que si se implementara una solución final por parte de la administración, la baja visibilidad que da un mal SEO se vería compensada con toda la red de páginas web del aparato administrativo.

4.2 Diseño detallado

En esta sección se explica el funcionamiento de la aplicación web partiendo del mapa de navegación. Después se explica el tratamiento que hace el servidor cuando le llega una llamada y por último se explica el modelo de datos. Se presentan muestras del código fuente para ejemplificar algunos componentes explicados.

4.2.1 Mapa de navegación

En la figura 9 se puede observar mapa de navegación de la aplicación. Este esquema es por el que navegará el usuario.

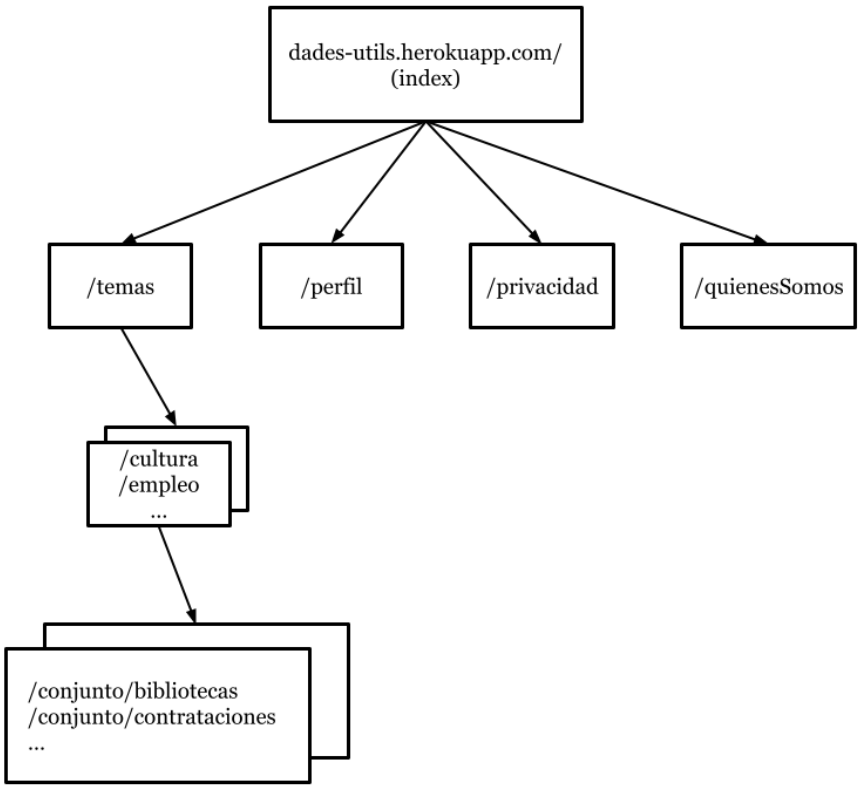

Figura 9: Mapa de navegación de la aplicación.

Dades Útils será el nombre de la aplicación y "herokuapp.com" es un sufijo que pone la plataforma de despliegue obligatoriamente. La URL básica redireccionará directamente a la página de temas. Se explica a continuación cada una de las páginas.

- **Menú principal** (/temas): Aquí se mostrarán varios temas que agrupan varios conjuntos de datos cada uno. Además debería haber un buscador que permita al usuario ir directamente al conjunto de datos que desee. En caso de que el usuario escoja algún tema, la página le llevará a la lista de conjuntos de datos que hay en ese tema. También se incluirá aquí un acceso directo a las vistas de datos que el usuario haya marcado previamente, lo que se llamará "suscripciones".

- **Menú de tema** (/cultura, /empleo etc.): Se mostrará una lista con todos los conjuntos de datos que hay disponibles y agrupados en este tema en concreto. Cada conjunto será un enlace a su vista de datos.
- **Vista de datos** (/conjunto/bibliotecas, /conjunto/contrataciones): Estas son las páginas donde se muestra la vista de los datos obtenidos directamente de las APIs de datos abiertos de la GVA. Se mostrarán los datos de una forma fácil de comprender y consumir por parte del usuario. Además, estas páginas tendrán que cumplir con las condiciones conocidas en el análisis legal.
- **Perfil** (/perfil): Tendrá un formulario para rellenar datos que luego puedan ser utilizados para darle una vista de los datos más personalizada. Además en esta página tendrá también su lista de conjuntos seleccionados y podrá eliminar su suscripción de cada uno de ellos. Estos conjuntos están accesibles desde otras partes de la aplicación para no tener que buscarlos cada vez que se quiera acceder a ellos.
- **Presentación** (/quienesSomos): Será una página con una breve explicación de qué es Dades Útils.
- **Privacidad** (/privacidad): Tendrá un texto explicando el tratamiento de los datos y demás cuestiones legales. A esta página se redirigirá cualquier enlace de política de privacidad. Además el texto mostrado cumplirá con los requisitos extraídos del previo análisis legal.

4.2.2 Recepción de llamadas por parte del servidor

En la Figura 10 se puede observar un diagrama que muestra los caminos que puede tomar una llamada de cliente, ya sea un navegador web solicitando la página o la propia página haciendo alguna llamada para consultar datos o cualquier otra funcionalidad.

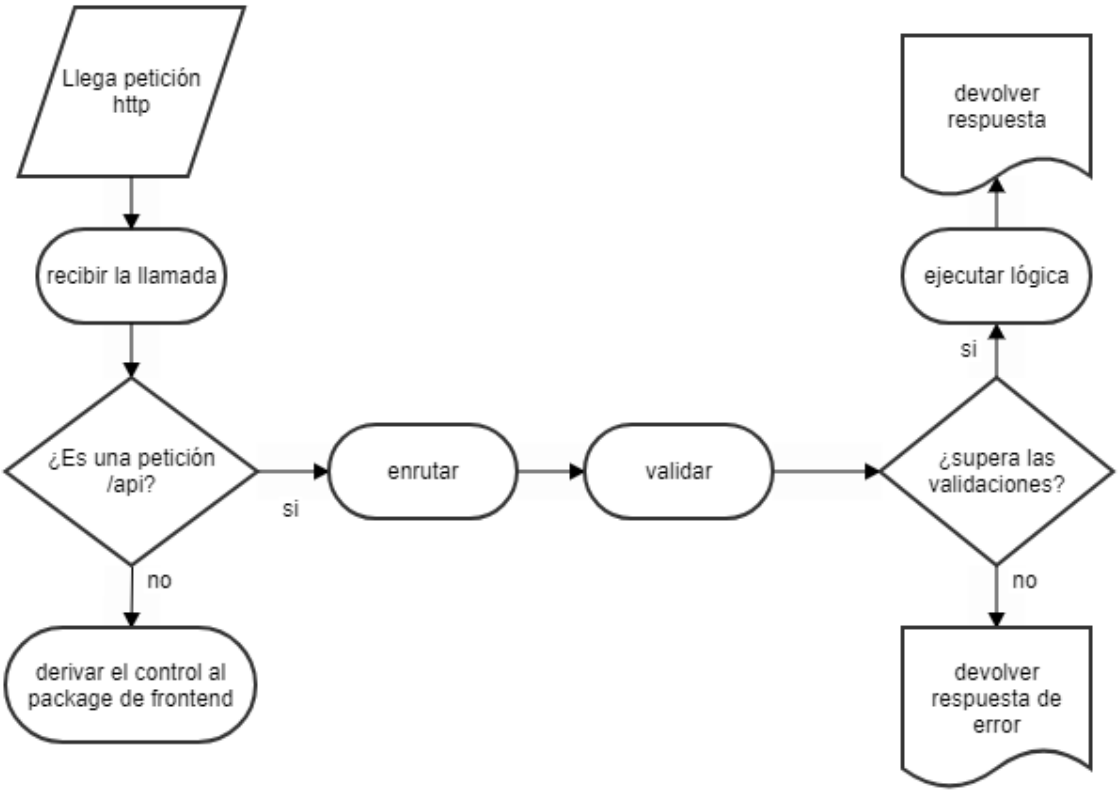

Figura 10: Diagrama de flujo de una petición http que llega al servidor.

Dependiendo del tipo de petición que se reciba, esta pasará por unos procesos u otros. Se detallan a continuación:

- **Recibir la llamada**: Se hará una primera separación entre las peticiones de navegadores que solicitan una de las páginas del lado del cliente y las peticiones de la propia página que llama al API del servidor con múltiples propósitos que se explicarán más adelante. A continuación, en la Figura 11 se muestran las líneas en las que ocurre.

```
26  //Rutas api
27  app.use( '/api', require('./routes/api') );
28
29  //Las demás rutas
30  app.get('*', (req, res) => {
31      res.sendFile(path.resolve(__dirname, 'public/index.html'))
32  })
```

Figura 11: Código fuente en el que se redireccionan las llamadas recibidas en el servidor.

La línea 27 recibe las llamadas a la API y las redirecciona a los enrutadores.
En caso de peticiones de navegadores, las líneas 30, 31 y 32 las reciben y las redirigen a la carpeta donde esté desplegado el lado del cliente.

- **Derivar a package frontend**: Esto es el despliegue del lado del cliente. Cuando una petición llegue aquí, se cargará la página que corresponda. En la Figura 12 se muestra parte del código fuente que se encarga de esto.

```
10  {
11    path: 'temas',
12    loadChildren: () => import('./subjectMenus/subject-menus.module')
13    .then(m => m.ThemesModule)
14  },
15  {
16    path: 'conjunto',
17    loadChildren: () => import('./datasets-pages/datasets-pages.module')
18    .then(m => m.DatasetsPagesModule)
19  },
```

Figura 12: Código fuente enrutamiento de menú principal y menú de tema.

Además debería tener un mecanismo para que en caso de que se solicite una página que no exista, la que se devuelva sea la del menú principal. En el pequeño extracto de código fuente de la Figura 13 se muestra cómo se implementa esto en Angular.

```
32  {
33    path: '**',
34    redirectTo: 'temas'
35  }
```

Figura 13: Código fuente enrutamiento de peticiones por defecto.

- **Enrutar**: En caso de que la petición sea al API del servidor, ya sea para recibir datos, suscribir un correo o cualquier otra cosa, lo primero que se hará será validarla. En función de la URL recibida se aplicarán unas validaciones u otras. Qué validaciones se deben aplicar y el controlador que ejecutará la lógica en caso de que se superen es algo que dictará el enrutador.
- **Validar**: Para cada llamada al API que se reciba habrá una lista de validadores determinados por el enrutador. Se encargarán de frenar cualquier llamada que no cumpla con unos requisitos mínimos referentes a los campos o cabeceras. Solo las llamadas que no requieran más información que la propia URL podrán progresar sin pasar por las validaciones. En la Figura 14 se muestran las validaciones aplicadas a la llamada que solicita la información a mostrar de un conjunto de datos o dataset.

```
24  router.get('/getDatasetData', [
25      check('datasetname', 'Campo datasetname vacio').not().isEmpty(),
26      check('datasetname').custom(datasetname =>{
27          if(!datasetExists(datasetname)) {
28              throw new Error('El dataset requerido no existe');
29          }else return true;
30      }),
31      validarCampos
32  ], getDatasetData);
```

Figura 14: Validación de una llamada de cliente.

Las líneas 25 a 30 son las validaciones. En la 25 se comprueba que el campo que debe indicar el dataset no está vacío. En las siguientes que el dataset al que se refiere existe.
Si alguna validación no se supera se devolverá una respuesta de error explicando cual ha sido el motivo de que no se superen las validaciones. En el ejemplo se pueden observar los mensajes "Campo datasetname vacío" y "El dataset requerido no existe". Si todas las validaciones se superan se pasará la llamada a los controladores para que ejecuten la lógica correspondiente a esa llamada. En el ejemplo ocurre en la línea 32.

- **Ejecutar lógica**: Se encargan los controladores y ejecutan la lógica necesaria para suplir la llamada. Serán los que utilicen los recursos no relacionados con el flujo de control de la llamada como: base de datos, APIs externas etc. También procesarán las respuestas de las APIs externas. Habrá varios controladores, aunque lo normal será que haya uno por URL de la llamada. Será el enrutador el que dicte qué controlador se hará cargo de cada llamada. Cuando se genere la respuesta, esta se devolverá. A continuación, en la Figura 15 se muestra un ejemplo del código fuente de uno de estos controladores.

```
10  const getDatasetList = async(req, res = response) => {
11      const {subject} = req.headers;
12      const datasets = await DatasetsDB
13      .find({subjects: subject}, 'title description selector')
14
15      return res.status(201).json({
16                          ok: true,
17                          datasets: datasets
18                      });
19  }
```

Figura 15: Código fuente del controlador para las peticiones de la lista de datasets de un tema.

Se encarga de suplir las llamadas que piden la lista de todos los datasets de un tema (subject) en concreto. En la línea 13 se extrae de la base de datos la información requerida. Y en las líneas 15 a 18 se devuelve una respuesta con esta información.

Para las llamadas que requieran una lógica especialmente larga o compleja se separa el código poniéndolo en un controlador complementario al que el controlador normal recurre para que ejecute esa lógica más compleja.

4.2.3 Modelos de datos

Habrá una colección en la base de datos para guardar la información sobre los conjuntos de datos llamada "datasets". Gracias a la tecnología Mongoose definir el modelo de la colección es muy sencillo. A continuación, en la Figura 16 se muestra el código fuente del modelado con Mongoose.

```
3   const DatasetSchema = Schema({
4       name: {
5           type: String,
6           required: true
7       },
8       lastRevision: {
9           type: Date,
10          required: true
11      },
12      lastGVAUpdate: {
13          type: Date,
14          required: true
15      },
16      idResources: {
17          type: [String],
18          required: true,
19          unique: true
20      }
21  });
```

Figura 16: Modelado de la colección datasets de la base de datos con Mongoose

Se explican a continuación las propiedades definidas en el modelo:

- **Name**: Una cadena de texto. Es el nombre del dataset dentro de la aplicación web. En las llamadas que haga el lado del cliente a la API del servidor en que tenga que identificar un conjunto de datos utilizará esta propiedad para que el servidor sepa a que conjunto se refiere.
- **LastRevision**: Una fecha requerida. Será la fecha de la última vez que se revisó si había datos nuevos publicados por la GVA de este conjunto de datos. Se actualizará con las revisiones periódicas automáticas que se harán para comprobar si hay nuevos datos.
- **LastGVAUpdate**: Una fecha requerida. Sera la fecha de la última actualización por parte de la GVA de los datos de este conjunto. A raíz del análisis legal esta información es necesaria pues en cada vista de datos se tiene que indicar la fecha en la que fueron actualizados.

- **IdResources**: Una lista de cadenas de texto requerida y única. Acumula los identificadores de los recursos de la GVA contra los que hay que enviar las llamadas a las APIs de consulta de datos.

4.3 Tecnologías que se utilizarán

En esta sección de listan las principales tecnologías utilizadas durante el desarrollo. Se dividirá en 4 apartados: herramientas para el desarrollo, tecnologías del lado del servidor, tecnologías del lado del cliente y tecnologías complementarias.

4.3.1 Herramientas para el desarrollo

Visual Studio Code[5]

Un editor de código fuente desarrollado por Microsoft muy popular. Tiene la posibilidad de personalizar muchísimo el entorno debido a su grandísima lista de plugins disponibles en el gestor de plugins integrado en la herramienta. Además tiene una excelente integración con Git, otra de las tecnologías que se van a utilizar en el proyecto. Una terminal integrada permite navegar por el directorio del proyecto, utilizar Git, npm, node etc.

Git[6]

Software de control de versiones que mantiene un registro de los cambios realizados en los diferentes archivos del proyecto. Permite la creación de ramas para separar las diferentes versiones, lo cual ayuda a proteger el código desplegado. Esto es muy conveniente cuando se integra con una función de despliegue automático, como es el caso en este proyecto.

[5] https://code.visualstudio.com/

[6] https://git-scm.com/

GitHub[7]

Es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control de versiones Git anteriormente mencionado. Es la plataforma más importante para proyectos Open Source. Este proyecto no es colaborativo pero se beneficiará de su cómoda interfaz gráfica y facilitará mucho el uso de algunas funcionalidades de Git como las pull requests y sus merges correspondientes. Además, es muy útil para consultar muy cómodamente el código de versiones anteriores.

Nodemon[8]

Nodemon permite lanzar rápidamente la versión de desarrollo de la aplicación Node. Además implementa la recarga automática de los cambios que hagamos en el código. Esto es que la aplicación se despliega en modo desarrollo cada vez que guardamos algún cambio en algún archivo. Gracias a esto la programación del código fuente y el debugging se hace mucho más fácil y rápida.

MongoDB Compass[9]

MongoDB Compass es una interfaz gráfica de usuario para la gestión de las bases de datos de MongoDB. Es muy útil para hacer consultas y modificaciones de los datos y un gran apoyo a la hora de hacer debugging y pruebas de una funcionalidad relacionada con la base de datos.

---

[7] https://github.com/

[8] https://www.npmjs.com/package/nodemon

[9] https://www.mongodb.com/es/products/compass

Postman[10]

Es una herramienta que se utiliza sobre todo para el testing y la exploración de API REST. Se utiliza en este proyecto para entender el funcionamiento de las diversas APIs que se van a emplear y para hacer pruebas con el backend. Sobre todo para las llamadas a la API del servidor. Esta herramienta permite configurar el método de la llamada, las cabeceras, el body y todo lo que pueda hacer falta. Así como consultar la respuesta devuelta por el servidor.

Angular CLI[11]

Es una interfaz en línea de comandos que nos aporta una gran comodidad para manejar angular desde consola. Facilita gestiones como crear componentes, servicios, módulos, despliegues etc.

4.3.2 Tecnologías para el lado del servidor

MongoDB[12]

Es una base de datos NoSQL, probablemente una de las más populares dentro de esta categoría. En MongoDB creamos un clúster que puede tener varias bases de datos. MongoDB guarda los datos en documentos almacenados en BSON, una versión binaria de JSON. Sin embargo, gracias a MongoDB Compass, podremos visualizar los datos de forma legible. La flexibilidad de una base de datos basada en documentos que no tienen un esquema fijo facilita el desarrollo. Sumando eso a que este proyecto utiliza un modelo de datos muy sencillo, MongoDB se presenta como la opción más adecuada.

[10] https://www.postman.com/

[11] https://angular.io/cli

[12] https://www.mongodb.com/es

Mongoose[13]

Es una librería para Node.js que nos permite escribir consultas para la base de datos con características como validaciones, construcción de queries, middlewares, conversión de tipos etc. Además nos permite definir un esquema que sirve para modelar la base de datos sin perder la flexibilidad de que esté basada en documentos.

Node.js[14]

Es un entorno de ejecución de javascript orientado a eventos asíncronos. De esta manera es ligero y eficiente. Además Node está diseñado para que no exista el bloqueo en la entrada de las peticiones y la salida de las respuestas. Aunque este proyecto no va a tener una gran envergadura, si la solución se aplicase por parte de las administraciones para todos sus conjuntos de datos y para el volumen de tráfico que se generaría, Node.js sería una buena opción por lo escalable que es.

Express[15]

Es el framework web más popular para Node.js. Express nos permite dirigir las peticiones http a sus respectivos controladores, facilita la generación de las respuestas, aplicar middlewares cuando se reciben peticiones http etc.

---

[13] https://mongoosejs.com/

[14] https://nodejs.org/es/

[15] https://expressjs.com/es/

API de Dades Obertes[16]

Esta es la API que la GVA habilita para realizar consultas a sus datos publicados. Se puede consultar el catálogo de datos para obtener información sobre los diferentes conjuntos de datos que se han publicado en el portal o se pueden hacer consultas directamente a los propios recursos que hay publicados. Además es posible hacer consultas mediante SQL, lo que podría ayudar a aligerar las respuestas.

CKAN[17]

CKAN es una muy popular API de gestión de recursos abiertos y es la empleada por el portal Dades Obertes de la GVA. De modo que para poder hacer consultas a los recursos del portal será necesario utilizar esta API. Tiene una clara y extensa documentación con ejemplos que facilitan el aprendizaje.

4.3.3 Tecnologías para el lado del cliente

Angular[18]

Un framework de código abierto puntero y muy popular actualmente. Desarrollado por Google para facilitar la programación de Single Page Applications (SPA), paginas donde un mismo archivo html sirve todo el contenido. Aun con esta arquitectura no se pierde la navegación por URL ni se genera una carga inicial inasumible, pues cuenta con mecanismos para la carga perezosa de los contenidos. Permite una organización modular y escalable del código. Se programa con Typescript.

[16] https://portaldadesobertes.gva.es/va/informacio-per-a-reutilitzadors

[17] https://ckan.org/

[18] https://angular.io/

Bootstrap[19]

Es un framework CSS que nos permite crear un diseño moderno y responsive desde el archivo html. Funciona mediante una extensa librería de clases que podemos aplicar a los elementos html. Les aporta las características de los estilos y la funcionalidad javascript que trae Bootstrap. Una de las características más importantes y populares de Bootstrap es la organización por filas y columnas.

Mapbox[20]

Un servicio de mapas similar al archiconocido Google Maps pero con una versión gratuita limitada. Se utilizará para representar en un mapa los datos en los que convenga ese tipo de visualización. Además tiene también un servicio de Geocoding, es decir, permite obtener coordenadas a partir de direcciones postales. Estas coordenadas se utilizarán para localizar los puntos concretos en el mapa.

4.3.4 Tecnologías complementarias

Heroku[21]

Es una plataforma como servicio (PaaS) donde se puede desplegar la aplicación. Tiene una versión gratuita limitada que cumple con creces las necesidades de este proyecto. Además permite la integración con GitHub para realizar despliegues automáticos.

---

[19] https://getbootstrap.com/

[20] https://www.mapbox.com/

[21] https://www.heroku.com/

5. Desarrollo de la solución propuesta

En esta sección se explica cómo se ha llevado a cabo el desarrollo de lo diseñado en apartados anteriores. Empezando por cómo se ha aplicado el enfoque ágil del proyecto, después se presentan los principales problemas y dificultades que han surgido a lo largo del desarrollo y las soluciones o decisiones en que han acabado.

5.1 Aplicación de la metodología

Esta sección está dividida en dos subapartados. En el primero se tratan las prácticas agiles aplicadas y como se han llevado a cabo. En el segundo se tratan las fases por las que ha pasado cada ítem desde su concepción hasta tenerlo implementado y probado.

5.1.1 Prácticas de metodología ágil

Tal y como se ha mencionado en el apartado 1.4.1, en este proyecto se han escogido una serie de prácticas de la metodología ágil de desarrollo de entre todas las disponibles en el Agilev Roadmap. En este apartado se explica cuáles concretamente se han intentado aplicar y cómo se ha hecho.

- PRA01 - Promover la sencillez en todos los aspectos. Ofrecer la solución más simple y mínima que pueda ser satisfactoria para el cliente. Estrategia MVP.
- PRA04 - Realizar planificación (y replanificación) frecuentemente (frecuencia de pocas semanas no meses).
- PRA08 - Poner el foco en conseguir un buen flujo de trabajo terminado.
- PRA09 - Gestión continua y multicriterio del trabajo pendiente para que esté siempre debidamente priorizado. Gestión del Backlog.

Para priorizar se ha utilizado el Kano Model [6], un sistema de clasificación de características de producto definido en 1984 por el Dr. Noriaki Kano. El sistema se basa en clasificar las características en 5 categorías muy bien diferenciadas. Brevemente las categorías son:

- **Básicas:** Son necesarias para la aplicación. El usuario las da por hecho.
- **De entusiasmo:** No son necesarias pero aportan valor y satisfacción en caso de implementarlas.
- **De resultado:** Son las que el usuario suele comparar de unas aplicaciones a otras. Producen satisfacción si están implementadas e insatisfacción si no lo están
- **Indiferentes**: No causan satisfacción si se implementan ni se echan en falta si no se implementan.
- **Inversas:** Al igual que las indiferentes, conviene evitarlas. En el caso de las inversas, causan insatisfacción si sí que se implementan.

Antes de empezar el desarrollo se generó un generoso backlog poblado con ítems correspondientes a desarrollos y algunas tareas necesarias que no trataban de escribir

código fuente. Acto seguido se priorizaron todas las tareas. La Figura 17 muestra el porcentaje de cada una de las categorías.

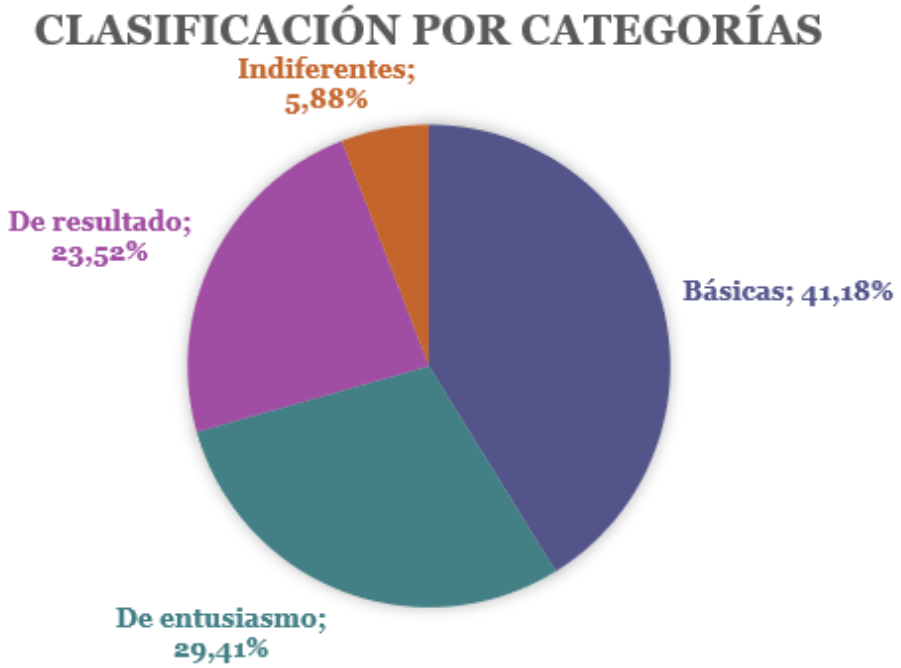

Figura 17: Gráfico de la clasificación de los ítems del backlog tras ser priorizados por primera vez.

Un porcentaje tan alto de ítems básicos resultó inesperado pero tendría su explicación más adelante. Siguiendo con la práctica 01, los ítems de la categoría "indiferente" se archivaron. Esto quiere decir que ya no aparecerían en el producto backlog facilitando así su gestión pero tampoco se perderían, permitiendo su consulta o incluso recuperación.

Siguiendo las prácticas 04 y 09 se ha hecho una revisión rápida del backlog con cada ítem terminado. En las primeras revisiones hubo una situación recurrente. Aunque se habían completado ítems de la categoría de básicos el porcentaje de básicos no bajaba, sino que se mantenía o incluso subía. Esto era debido al desconocimiento de las tecnologías y del ámbito de las aplicaciones web en general. En muchas ocasiones, a la hora de enfrentar un ítem del backlog este acababa siendo mucho más grande de lo estimado en un principio. Esto llevaba a que tener que dividirlo en varios ítems más pequeños pero con el mismo nivel de prioridad casi siempre.

La línea de actuación ante esto fue hacer una revisión más realista de los ítems y planificar como se tenía que llevar a cabo cada uno. De esta manera se pudieron identificar ítems épicos y dividirlos antes de tener que afrontarlos. Esta división llevó a un gran incremento del porcentaje de ítems o tareas básicas por lo que se decidió realizar una revisión de las prioridades. Esta vez, con algo más de experiencia, se decidió que en vez de tener como objetivo presentar la mayor cantidad de vistas de conjuntos de datos el objetivo sería presentar una de ellas con mayor calidad, dejar otra más para Entusiasmo y archivar las demás.

- PRA02 - Abordar trabajo de forma incremental.
- PRA03 - Realizar entregas frecuentes de unidades de trabajo terminadas.
- PRA40 - Integrar de forma continua en el producto el trabajo terminado.

La práctica 02 ha sido una de las que más ha costado llevar a cabo. A partir aproximadamente de la mitad del desarrollo ya se pudo hacer más eficazmente. Sin embargo la gran cantidad de tareas de tipo básico hacían muy difícil tener un despliegue funcional desde el principio.

Las practicas 03 y 40 también fueron laboriosas pero se acabaron pudiendo implementar perfectamente. Gracias a las prácticas 07, 10 y 36 que se explicarán a continuación el flujo de ítems terminados era bastante constante. Y gracias a la integración de GitHub con Heroku la práctica 40 se pudo llevar a cabo fácilmente. Cabe decir que para configurar esta integración de una forma en que fuera cómodo trabajar con ella hubo que superar unas dificultades que se explicarán más adelante en este trabajo.

- PRA07 - Evitar invertir esfuerzo en adelantar trabajo que no esté comprometido y/o no esté cercano a su entrega.
- PRA10 - Limitar el trabajo en proceso (WIP) es decir la cantidad de unidades de trabajo que tiene el equipo en una determinada actividad
- PRA36 - Reducir las interrupciones o cambios de contexto que afectan en su trabajo a los miembros del equipo.

Llevar a cabo estas prácticas ha sido muy sencillo. Solo al principio no se cumplía la práctica 07 pero en cuanto se tuvo que hacer la revisión mencionada anteriormente en la que se dividieron tareas grandes en varias más pequeñas, el riesgo de que no se pudieran cumplir los plazos ni para las tareas más básicas se hizo evidente. A partir de ese momento solo hubo uno o dos ítems, como máximo, circulando por el tablero Kanban a la vez.

- PRA05 - Acotar el trabajo previsto para un periodo en base a su estimación y la correspondiente coherencia con la capacidad del equipo.

Esta práctica se aplicó, aunque sufrió una modificación tras las primeras revisiones del backlog. Los periodos se eliminaron porque no aportaban gran valor teniendo en cuenta la integración continua y el hecho de que no hubiera esprints. Sin embargo la estimación fue muy útil para tener una noción de la presión que había en cada momento y sobre todo si había que intentar aumentar la capacidad por ejemplo invirtiendo más tiempo. Muy pronto se hizo evidente una tendencia a estimar a la baja. Para solucionar esto a partir de ese momento se hacía una estimación en dos pasos para cada ítem. El primer paso era hacer una estimación normal y corriente como se llevaba haciendo hasta ahora. El segundo paso era multiplicar esa estimación por 1.5 para intentar tener un valor que hubiese tenido en cuenta la tendencia personal. Funcionó, aunque hubo un par de variaciones en la multiplicación a la largo del proyecto. También cabe decir que hacia el final, las estimaciones salían muy naturalmente ya aumentadas para corregir la tendencia natural.

- PRA28 - Documentar, pero solo lo estrictamente necesario. Que sea rentable el aprovechamiento de la documentación respecto del esfuerzo asociado a elaborarla.
- PRA42 - Mejorar continuamente la organización interna del producto para facilitar su mantenimiento. Refactoring.

La documentación ha sido mínima. Se ha limitado a la especificación mínima para el testing, algún documento donde explicar algunos conceptos que era importante que mantuviesen su consistencia a lo largo del desarrollo y breves comentarios en el código.

No se ha hecho una gran cantidad de refactoring pero los tipos que se han llevado a cabo varias veces han sido la extracción de funciones y la eliminación de comentarios mediante el renombrado de variables y métodos.

- PRA27 - Trabajo centrado en satisfacer pruebas de aceptación acordadas con el Product Owner.

Aunque no hubiera un Product Owner que pudiera darles el visto bueno a las pruebas, esta práctica se ha aplicado de forma constante a lo largo del proyecto. Algunos ejemplos de pruebas de aceptación son:

<<Acceder a una url no prevista>>
Condición Ninguna
Pasos El usuario introduce una url que no es una ruta real contemplada por el servidor.
Resultado esperado La aplicación le redirige automáticamente al menú principal

<<Buscar un conjunto de datos en el buscador con éxito>>
Condición Que existan uno o más resultados para la búsqueda del usuario.
Pasos El usuario introduce un término en el buscador de conjuntos.
Resultado esperado Se despliega una lista con enlaces a los diferentes resultados de la búsqueda del usuario.

<<Navegar el menú de un tema en concreto>>
Condición

El usuario se encuentra en el menú principal.
Pasos
El usuario selecciona un tema del menú principal.
Resultado esperado
La aplicación le redirige a la página del tema en concreto, donde se muestra una lista paginada con los conjuntos de datos de ese tema.

<<Desplegar la segunda capa informativa de la política de privacidad>>
Condición
El usuario se encuentra en la página de la política de privacidad con la segunda capa informativa sin desplegar.
Pasos
El usuario hace clic en el enlace para desplegar la segunda capa informativa.
Resultado esperado
El texto de la segunda capa informativa aparece, el enlace pulsado por el usuario desaparece y aparece otro para dejar de mostrar la segunda capa.

<<Navegar a la vista de un conjunto de datos>>
Condición
Ninguna
Pasos
El usuario selecciona un conjunto de datos de la lista
Resultado esperado
La aplicación redirige al usuario a la vista de los datos de ese conjunto de datos, donde aparece un cartel bloqueando la aplicación hasta que se obtengan y presenten los datos. Una vez obtenidos y procesados el cartel desaparece y los datos se muestran.

- PRA29 - Establecer pautas para gestionar convenientemente el retrabajo.

Como ya se preveía en el capítulo en que se presentaban estas prácticas por primera vez, donde más retrabajo se ha tenido que hacer es en adaptar el código a la arquitectura. Un claro ejemplo sería que al principio la información de cada conjunto de datos se almacenaba en un archivo local entre el código fuente del frontEnd. Pero conforme se implementaron las funcionalidades relacionadas con la base de datos esta decisión se ha revelado inadecuada. En ese momento se integró esta información de cada dataset con la que ya estaba plasmada en la base de datos. Los atributos que se añadieron a la colección “datasets” de la base de datos fueron los siguientes:

- **Title**: Una cadena de texto requerida. Será el nombre con el que aparezca este conjunto de datos en las interfaces del lado del cliente.
- **Description**; Una cadena de texto requerida. Una breve descripción que acompañará al título en las interfaces del lado del cliente.
- **Selector**: Una cadena de texto requerida y única. Esta cadena la empleará el lado del cliente para saber encontrar el componente que debe mostrar para el conjunto de datos al que quiere acceder el usuario.
- **Subjects**: Lista de cadenas de texto requerida. Indican los temas de los que este conjunto forma parte.

Pasando estos atributos a la base de datos se pudo eliminar el archivo donde se guardaban en el código fuente del lado del cliente. Otro beneficio que se obtuvo con esto fue mejorar mucho la escalabilidad pues se eliminaba una de las tareas que había que realizar cada vez que se quisiera añadir un conjunto de datos nuevo. Este proceso se explica más adelante. Cabe decir que la tarea de hacer que los componentes que necesitaban estos datos los cogieran de la base de datos fue una cantidad de retrabajo considerable.
Por suerte el haber escogido una base de datos basada en documentos significó que este cambio fuera menos laborioso.

5.1.2 Fases de cada ítem del backlog

Tal y como se ha mencionado en el apartado 1.4.2, cada ítem del backlog ha tenido un ciclo de vida que pasa por 4 fases. En este apartado se detalla cómo ha funcionado la aplicación de estas fases con ejemplos y estadísticas.

Fase 1 – Especificación de requisitos o definición de la tarea

La especificación de requisitos se ha realizado mediante pruebas de aceptación y descripciones breves, tal y como se planeó en la fase de diseño. Puesto que ya se han puesto varios ejemplos de pruebas de aceptación, se presentan a continuación algunas de las descripciones breves.

- **Llamadas SQL bibliotecas**: Implementar las llamadas SQL al api de datos de la GVA para obtener los datos de bibliotecas.
- **API CKAN**: Crear las funciones para llamar a la API de CKAN para obtener la información de los conjuntos.
- **Generar icono**: Crear el icono para la aplicación partiendo del logo
- **Endpoints suscripción**: implementar los endpoints de la funcionalidad de suscripción (check, get, update, delete, new)
- **Controlador especifico bibliotecas**: implementar el controlador específico del conjunto de datos de bibliotecas que saque los datos del API de la GVA y los procese.
- **Crear el servicio para llamar al API de Geocoding**: La API de Mapbox solo admite coordenadas. Hay que crear un servicio que reciba direcciones, las analice y llame al API de Geocoding.

- **Buscador de conjuntos:** Implementar el buscador de conjuntos del menú principal que genere una lista con enlaces a los conjuntos de datos que encajen con la búsqueda.

Fase 2 – Prototipado de las interfaces de usuario

Para este paso se ha utilizado la herramienta online Gliffy. Es de pago pero tiene una versión gratuita de prueba que llega a cumplir con las necesidades del proyecto. Además cuenta con bastante material para crear los prototipos aunque sean solo visuales y no funcionales. En las siguientes figuras se presentan los prototipos de la interfaz de usuario para el menú principal y el menú concreto de tema, tanto la versión para ordenador como la versión para teléfono móvil. En el caso del menú de tema se ha puesto el ejemplo como si fuera del tema “empleo” pero podría ser cualquier otro tema.

Figura 18: : Prototipo de la interfaz de usuario del menú principal, versión de ordenador.

Figura 19: Prototipo de la interfaz de usuario del menú principal, versión de teléfono móvil.

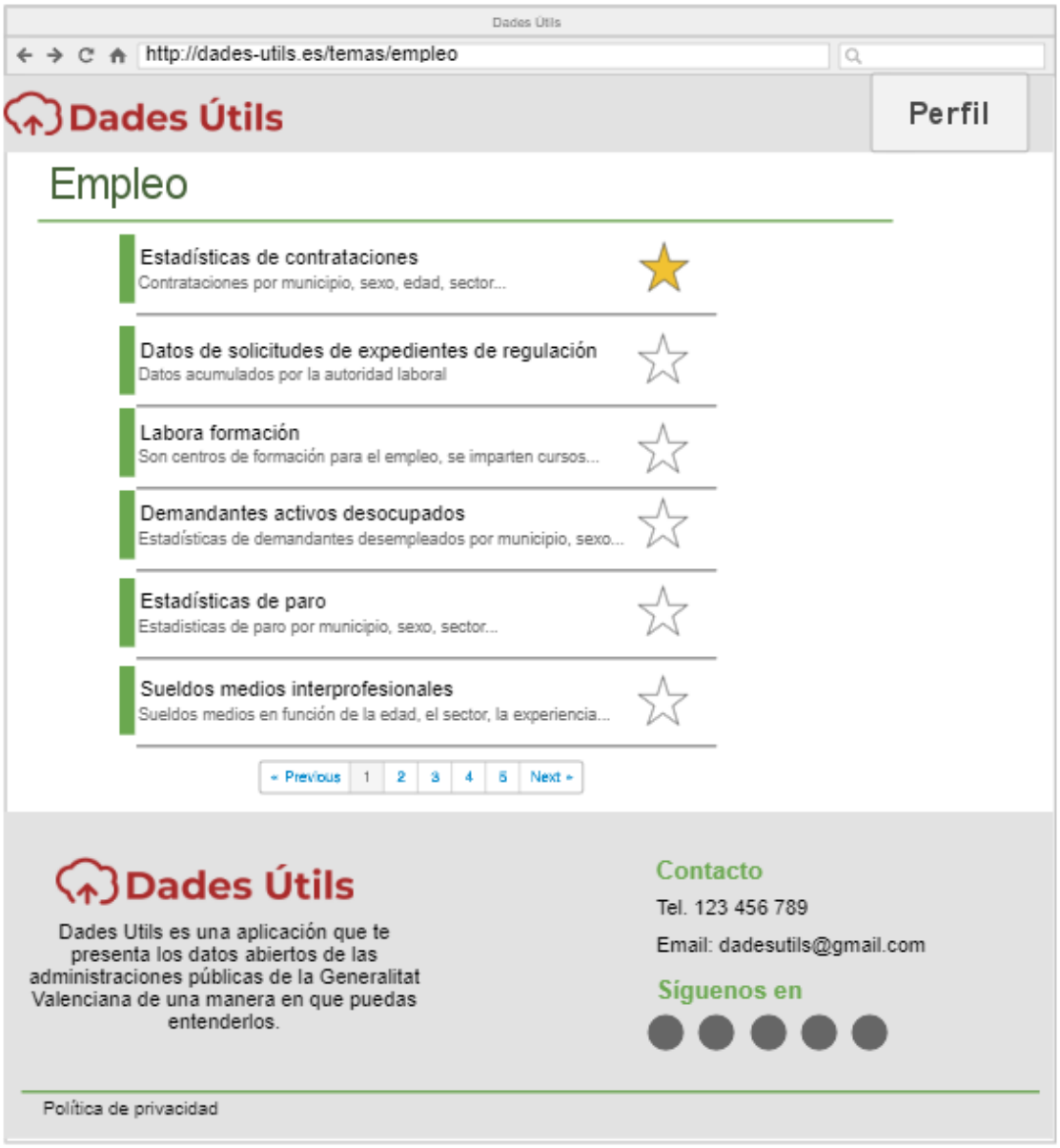

Figura 20: Prototipo de la interfaz de usuario del menú concreto de temática, versión de ordenador.

Figura 21: Prototipo de la interfaz de usuario del menú concreto de temática, versión de teléfono móvil.

Fase 3 - Implementación.

En esta fase se implementaba el código fuente requerido para la funcionalidad. Para las tareas del backlog que no implicasen alteraciones en el código fuente como por ejemplo: estudiar alguna API, buscar un servicio externo, reestructurar alguna parte del repositorio, configurar la integración continua automática etc. Esta fase significaba llevar a cabo la tarea, es decir, hacer todo lo necesario para que el trabajo quedara realizado y a la espera de las comprobaciones.

Fase 4 - Pruebas de sistema e integración.

En esta fase se realiza el testing de las funcionalidades desarrolladas. En línea con la práctica ágil 28, se ha generado una definición mínima de las pruebas. La documentación requerida para el testing solo era la definición de los posibles escenarios Entendiendo por escenario el estado en el que el sistema se encontrará en el momento en que la funcionalidad desarrollada se ejecuta. Algunos ejemplos de estas definiciones son:

- **Menú principal**
  - Se tienen solo 2 temas
  - Se tienen 10 temas
  - Se tienen 20 temas

- **Menú concreto de tema**
  - Se tiene 1 solo conjunto de datos
  - Se tienen 8 conjuntos de datos
  - Se tienen 10 conjuntos de datos
  - Se tienen 34 conjuntos de datos
- **Footer**
  - La página tiene 20.000px de altura
  - La página tiene 300px de altura
- **Solicitud API servidor pidiendo la lista de nombres de los conjuntos de un tema**
  - El tema solicitado no existe
  - El tema solicitado sí que existe pero no tiene conjuntos
  - El tema solicitado sí que existe y tiene 1 conjunto
  - El tema solicitado si existe y tiene 15 conjuntos

Para realizar el testing se exploraba toda la funcionalidad tanto en pantalla de ordenador como de teléfono móvil. Todo partiendo de cada uno de los escenarios y habiendo poblado la base de datos con información realista de relleno. En la Figura 20 se observa un gráfico que muestra la cantidad de errores encontrados cada vez que se hizo testing. Solo en el 10% de las veces no se encontró ningún error.

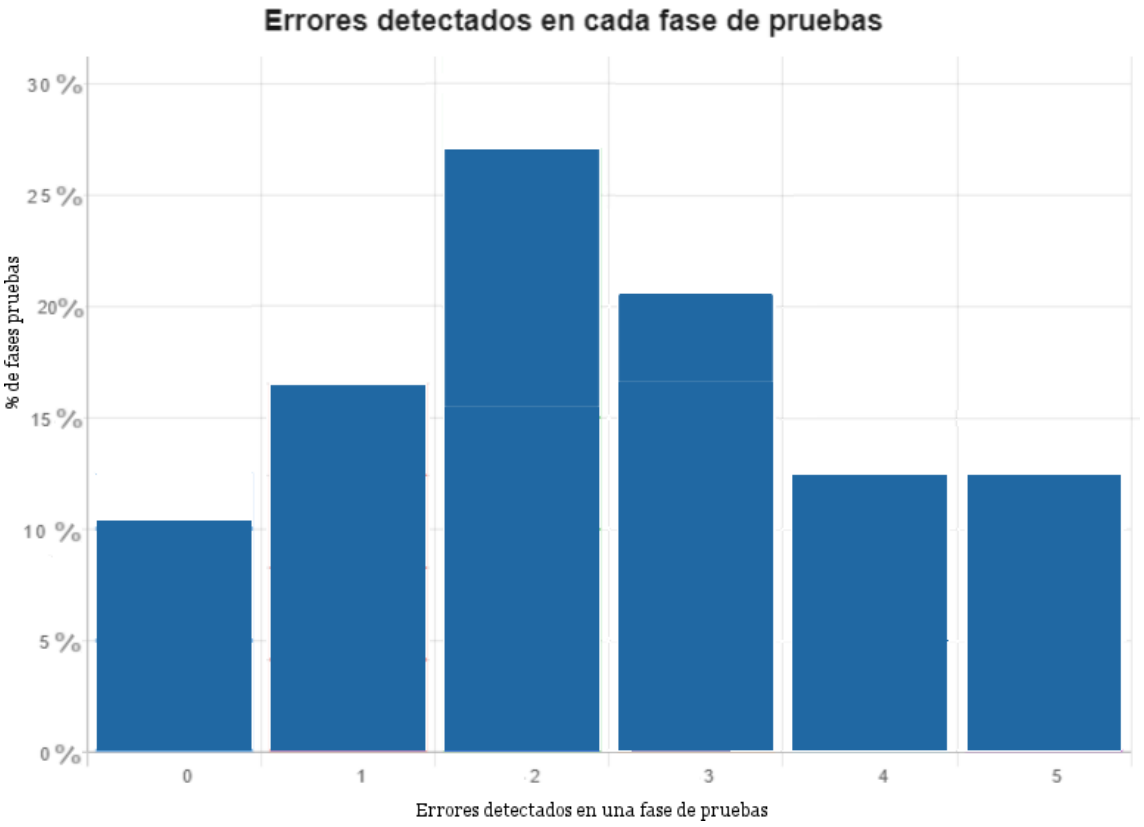

Figura 22: Gráfico de la cantidad de errores que se detectaron en las fases de pruebas.

Puede haber influido en la gráfica el que cuando se encontraba un error bloqueante o de fácil arreglo el testing se interrumpía para pasar a arreglar el error detectado de inmediato. Además, de las veces que no se encontraron fallos, solo se han incluido aquellas en las que era la primera vez en que el ítem a probar pasaba por la fase de pruebas.

Las pruebas de aceptación y el testing en producción no se superaban en una fracción prácticamente nula de las ocasiones.

Fase 5 – Evaluación de usabilidad

Estas pruebas se han realizado casi enteramente por videoconferencia. El usuario tenía que conseguir ciertos objetivos, que se pedían mientras compartía su pantalla. Además la videoconferencia estaba siendo grabada. Al finalizar se revisaba la grabación para cronometrar el tiempo que había necesitado el usuario para lograr los objetivos y el número de acciones innecesarias que había realizado. Las acciones innecesarias pueden ser navegar a una página que no está incluida en el camino hacia la información requerida, desplegar algún texto innecesario, realizar una búsqueda sin resultados o con resultados entre los que no se encuentra la información requerida etc. Al acabar la videollamada también se les enviaba un formulario elaborado con Google Forms para medir la satisfacción percibida por cada usuario mediante la escala SUS (System Usability Scale) [8]. Toda la información que se recogió en estas pruebas fue anónima.

A lo largo del proyecto se realizaron dos sesiones de pruebas con 5 usuarios realizando las pruebas en cada sesión. Ningún usuario de la primera sesión participó en la segunda. Estos son los datos de las pruebas:

Primer grupo de pruebas

- Objetivo: Localizar en el mapa la biblioteca llamada “Agencia De Lectura Gilet”.
  - Lograron el objetivo: 5/5 usuarios
  - Media de tiempo empleado: 45.78 segundos
  - Media de acciones innecesarias: 1.5 acciones
- Objetivo: Acceder a la lista de conjuntos de datos del tema “empleo”
  - Lograron el objetivo: 5/5 usuarios
  - Media de tiempo empleado: 5.5 segundos
  - Media de acciones innecesarias: 0.4 acciones
- Puntuación media obtenida de las encuestas SUS: 82(Notable)

Segundo grupo de pruebas

- Objetivo: Averiguar el email de la biblioteca llamada “Agencia De Lectura Gilet”
  - Lograron el objetivo: 5/5 usuarios
  - Media de tiempo empleado: 35.22 segundos
  - Media de acciones innecesarias: 1.5 acciones

- Objetivo: Averiguar quién es el responsable del tratamiento de los datos recogidos por Dades Útils
    - Lograron el objetivo: 5/5 usuarios
    - Media de tiempo empleado: 46.68 segundos
    - Media de acciones innecesarias: 2 acciones
- Objetivo: Acceder a la fuente de los datos de la página del conjunto de datos "Bibliotecas"
    - Lograron el objetivo: 5/5 usuarios
    - Media de tiempo empleado 47.10 segundos
    - Media de acciones innecesarias: 0.6 acciones
- Puntuación media obtenida de las encuestas SUS: 86(Notable)

5.2 Principales problemas y dificultades

Lo cierto es que el flujo de trabajo terminado ha sido constante para la mayoría de las tareas del backlog. Sin embargo, sí que han surgido problemas que han consumido una parte importante del tiempo. Se resumen en esta sección

5.2.1 CORS

La política Cross Origin Resource Sharing (CORS) es una medida de seguridad que evita que una página web cargue datos de un origen distinto a sí misma. Esto se aplica en el navegador, que bloquea las respuestas de las llamadas de este tipo de la página. Este problema surgió muy al principio del desarrollo del proyecto, en el momento de hacer una prueba de petición de datos por parte del lado del cliente a la API de datos de la GVA. Costó mucho entender el problema puesto que solo se dio en el momento en que la prueba se hizo desde una primera versión de la aplicación del lado del cliente. Anteriormente se habían hecho pruebas desde la herramienta Postman y no dio este bloqueo. El desenlace fue que el bloqueo no se daba en Postman porque la herramienta lleva integradas unas medidas para superar la política CORS. Y como solución en el proyecto se diseñó que sería el backend el encargado de hacer estas llamadas a servidores externos a petición del lado del cliente para luego procesar la respuesta y devolverle a este la información que requería.

5.2.2 Refactoring de código fuente a base de datos

Al principio se había implementado que los menús concretos de tema sacarían la lista de datasets de un fichero fuente. Sin embargo, conforme se fueron implementando más partes del código que requerían de esta misma información y de información de base de datos se hizo necesario trasladar toda la información de los datasets a base de datos. Esto implicó bastante retrabajo pues el código que se encargaba de sacar la información del fichero fuente se tuvo que eliminar e implementar todo el flujo de control para que el cliente hiciera una petición al servidor, este consultara los datos en base de datos los procesara y los devolviera al cliente. No fue especialmente complejo pero todo el tiempo que se invirtió en ello fue retrabajo.

5.2.3 Configurar el despliegue automático integrando Heroku con GitHub

Para seguir la práctica 40 de Integrar de forma continua lo más conveniente era utilizar la integración de Heroku con GitHub. Al configurarla, cada vez que se hacía una subida a la rama "deploy" del repositorio en GitHub, Heroku desplegaba automáticamente la nueva versión. El problema era que si mezclaba la rama de desarrollo con la rama de despliegue, la carpeta frontend donde estaba el código fuente sin compilar del lado del cliente también se añadía a la rama de despliegue. Y esto no era tolerable, porque Heroku esperaba una única carpeta en la rama. Tras estudiar un par de opciones con Git y cambiar una configuración de Angular, el proceso para hacer el despliegue e integrarlo en producción quedó así:

1. Mezclar la rama de la funcionalidad con la rama de desarrollo
2. Cambiar a la rama de desarrollo y compilar el frontend
3. Subir los cambios y cambiar a la rama de despliegue
4. Eliminar los módulos de Node.js que no tenían seguimiento del repositorio local
5. Cambiar la carpeta backend a la de la rama de desarrollo mientras la rama local sigue siendo la de despliegue
6. Subir los cambios a la rama de despliegue

Es probable que con un poco más de investigación el paso 4 se hubiese podido automatizar de alguna manera pero llegados a este punto el proceso se había reducido lo suficiente como para que fuera bastante rápido.

5.2.4 Llamadas SQL al api de datos de la GVA

En el portal de datos de la GVA hay una documentación básica para entender cómo funcionan las llamadas para consultar los recursos publicados por la GVA. Una de las funcionalidades disponibles es realizar una consulta SQL en la llamada. Esta hubiese sido una opción muy conveniente. Pues con SQL se podría optimizar la respuesta para recibir únicamente la información deseada y hacer un preprocesado de los datos. Sin embargo al hacer estas consultas no se consigue obtener ningún tipo de respuesta. Investigando la documentación de la API CKAN, que es la que ha aplicado la GVA para las consultas a sus datos se puede ver que las consultas SQL no vienen por defecto con la API sino que es necesario aplicar también una extensión. Esta extensión sirve para poder hacer consultas directamente a los recursos, con lo cual queda claro que la GVA la ha aplicado. Si no fuera así los demás tipos de consultas a los recursos tampoco funcionarían. Consultando un poco más en detalle la documentación de CKAN se puede leer que las consultas SQL en concreto no están disponibles en la versión "legacy" de esta extensión. Llegados a este punto solo queda confirmar que las consultas infructuosas de prueba estaban bien hechas. De esa manera quedaría claro que el problema está en que la extensión está desactualizada. Para ello se buscó otro API de alguna otra comunidad autónoma que aplicara la API CKAN. Se hizo la prueba con el de la ciudad de Barcelona y funcionó perfectamente. Quedaba así comprobado que la GVA tenía desactualizada la extensión necesaria para atender consultas SQL.

Había que buscar otra manera de consultar los recursos de la GVA. Al haber explorado tan en detalle la guía de CKAN, no requirió mucho tiempo encontrar esta otra manera.

5.2.5 Los recursos de la GVA

Otro problema vino con la política que tiene la GVA con respecto a las subidas de recursos a sus datos abiertos. Como se plasmó en el apartado 3.1 Modelo conceptual, el portal de datos de la GVA tiene, entre otros, los siguientes 2 elementos:

- Recurso: un archivo con los datos en bruto que se puede descargar y que es contra el que se lanzan las consultas que obtienen los datos.
- Conjunto: un grupo de 1 o más recursos estrechamente relacionados.

Un ejemplo podría ser:

- Conjunto: Estadísticas de contrataciones 2020
    - Recurso: Estadísticas de contrataciones 2020_enero
    - Recurso: Estadísticas de contrataciones 2020_febrero
    - Recurso: Estadísticas de contrataciones 2020_marzo
    - Recurso: Estadísticas de contrataciones 2020_abril
    - ...

En este caso hay un conjunto anual en el que se van subiendo recursos mensuales.

Si este (o cualquier otro) fuera el estándar a la hora de automatizar la actualización automática de los recursos en Dades Útils sería sencillo. Solo habría que identificar el patrón y desarrollar una automatización a medida.

Sin embargo, explorando los diferentes conjuntos se puede ver que no siempre se sigue esta política. Hay veces en que durante más de un año solo ha habido un conjunto con un recurso que se va actualizando. Otras veces por cada recurso que se quiere subir se crea un conjunto, ya sea un recurso anual o mensual. Incluso hay conjuntos en los que no siempre se ha seguido el mismo patrón. Empezando a subir un conjunto cada año con un recurso anual para luego cambiar a un recurso cada año con un recurso mensual.

Todo esto aumenta la complejidad de la automatización. La solución ha sido la siguiente.

En primer lugar se han identificado las diferentes políticas que hay en el portal para subir los recursos y se le ha dado a cada una un número en función del trabajo que hay que hacer para automatizar la actualización.

0. Para los recursos que no requieren actualización
1. Para los conjuntos que solo tienen un recurso que se va actualizando y con lo cual solo hay que actualizar la fecha en la que la GVA actualizó el recurso por última vez

2. Para los conjuntos en los que se va subiendo regularmente un recurso cada cierto tiempo. Y con lo cual hay que actualizar la fecha en la que la GVA subió un recurso por última vez y la lista de recursos de los que obtener información.
3. Para los irregulares. En esta categoría van los datos en los que se ha ido cambiando de política con el tiempo.

Estos números se guardan como un atributo más en la base de datos "datasets" con el nombre updateType, un número requerido. Todos los datasets de tipo 0, 1 o 2 son actualizados automáticamente por el mismo código. Para los de tipo 3 el servidor, de forma personalizada para cada conjunto, se encarga de hacer las peticiones adecuadas.

5.3 Cómo añadir una vista de datos

Para facilitar la comprensión del funcionamiento del proyecto se explican a continuación todo lo que habría que hacer para añadir una nueva vista de datos al portal. Gracias a la arquitectura del proyecto y a las automatizaciones el trabajo requerido se reduce bastante. Se puede dividir en 3 etapas análisis, servidor y cliente.

5.3.1 Trabajo de análisis

Lo primero que habrá que hacer es explorar el conjunto de datos escogido en el portal de la GVA para determinar qué política siguen sus actualizaciones. Con esta exploración se conseguirá clasificar el conjunto de datos en una de las 4 categorías explicadas en el apartado 5.2.5 de este trabajo. Es crucial identificar la categoría correctamente. De ello dependerá que el código que actualiza automáticamente los recursos e información del conjunto lo pueda hacer.

5.3.2 Trabajo en el servidor

Lo primero que se tendría que hacer en este paso es añadir una entrada en la base de datos con la información del nuevo conjunto. La entrada consta de:

- Un nombre simplificado a que se utilizará a nivel interno de la aplicación entre otras cosas para generar la url.
- Un título y una descripción que serán los que vea el usuario.
- La clasificación obtenida del paso anterior, el análisis en el portal de la GVA.
- El selector con el que el lado del cliente podrá encontrar la vista de datos que muestra este conjunto
- Los temas dentro de los cuales se clasifica este conjunto.

El resto de los campos de la base de datos se completarán con la primera actualización automática.

Lo siguiente será añadir código fuente en el servidor. El código necesario a implementar será el controlador. Cuando se recibe una llamada de un cliente que quiere mostrar los datos de un dataset, el controlador se encarga de obtenerlos del portal de la GVA. También los procesa para devolver al cliente solo los que necesita para su vista de datos. En caso de que la clasificación obtenida en el análisis del

apartado anterior haya sido 3. Esto implica que el portal de la GVA no sigue una política uniforme con respecto a las actualizaciones de este conjunto de datos. En este caso también será necesario añadir código en el actualizador automático de recursos, el componente que se encarga de revisar periódicamente si hay recursos nuevos disponibles en el portal para cada dataset.

5.3.3 Trabajo en el cliente

En el cliente lo único que se tendrá que hacer es implementar la vista de datos del nuevo dataset. Para esto convendría contar con algo de conocimiento experto sobre el dataset. De esta manera podríamos saber qué información es más probable que le sea útil al ciudadano. Será muy importante que se incluyan en la vista un par de componentes que ya están implementados y que gracias a Angular se pueden reutilizar. Estos componentes son los que muestran la fecha de actualización de los datos y los enlaces a las fuentes. Ambos requisitos legales impuestos por la GVA.

Una vez hecho todo esto ya solo faltará hacer un despliegue y la nueva vista de datos estará disponible en la aplicación.

6. Resultados, conclusiones y relación con los estudios

En esta sección se examinará si se han alcanzado o no los objetivos y las conclusiones que he sacado de todo el proyecto.

6.1 Resultados

A continuación se examinarán los objetivos uno por uno examinando si se han alcanzado o no y en qué medida.

- Mostrar datos obtenidos directamente de los publicados en el portal de datos de la GVA.

Este objetivo se ha cumplido por completo ya que del único origen del cual se obtienen los datos que luego se procesan para mostrarse al usuario son las APIs de consulta del portal de datos abiertos de la GVA. Una buena medida para demostrar esto sería dejar el repositorio abierto para que se pueda consultar el código fuente. De esta manera cualquiera podría examinarlo y ver que efectivamente los datos no se manipulan de ninguna manera. Sin embargo si se hiciera conforme está ahora el proyecto tokens de autentificación quedarían expuestos, con lo cual no se posible de momento.

- Que los datos se muestren de una manera cómoda, fácil de entender y útil por parte de los ciudadanos.

Este objetivo también ha quedado cumplido pues las pruebas con usuarios han tenido muy buenas cifras. Cabe destacar la calificación de Notable en las encuestas con escala SUS así como el hecho de que todos y cada uno de los usuarios testados hayan logrado todos los objetivos que se les plantearon.

- Ofrecer un diseño responsive que se adapte a los diferentes tamaños de pantalla.

Es sobre todo gracias a Bootstrap que este objetivo se ha podido cumplir tan satisfactoriamente y de una manera tan cómoda. Para evidenciarlo, en las siguientes figuras se pueden apreciar capturas del resultado final en la versión para ordenador y para teléfono móvil.

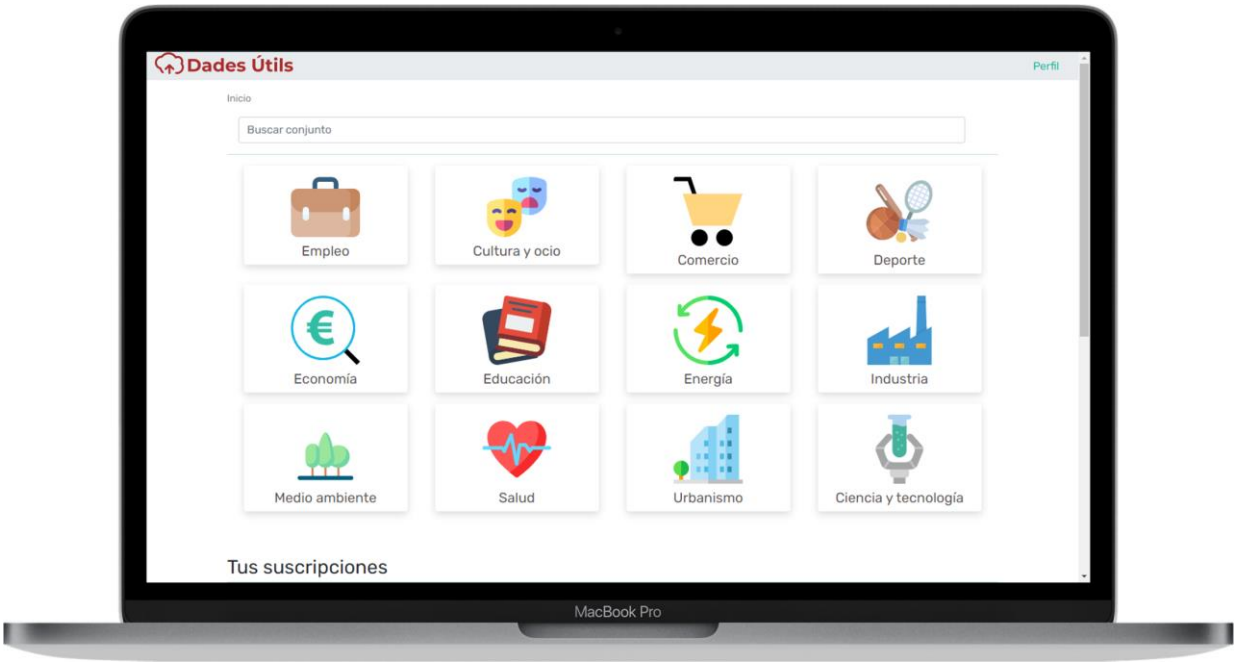

Figura 23: Captura de la parte superior del menú principal, versión de ordenador.

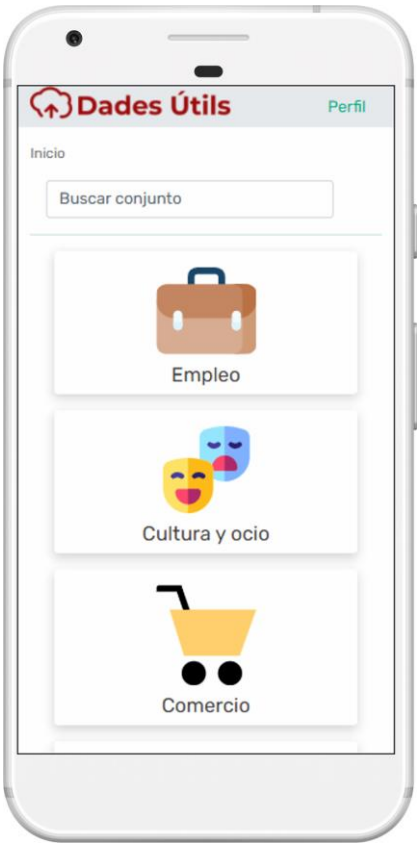

Figura 24: Captura de la parte superior del menú principal, versión de teléfono móvil.

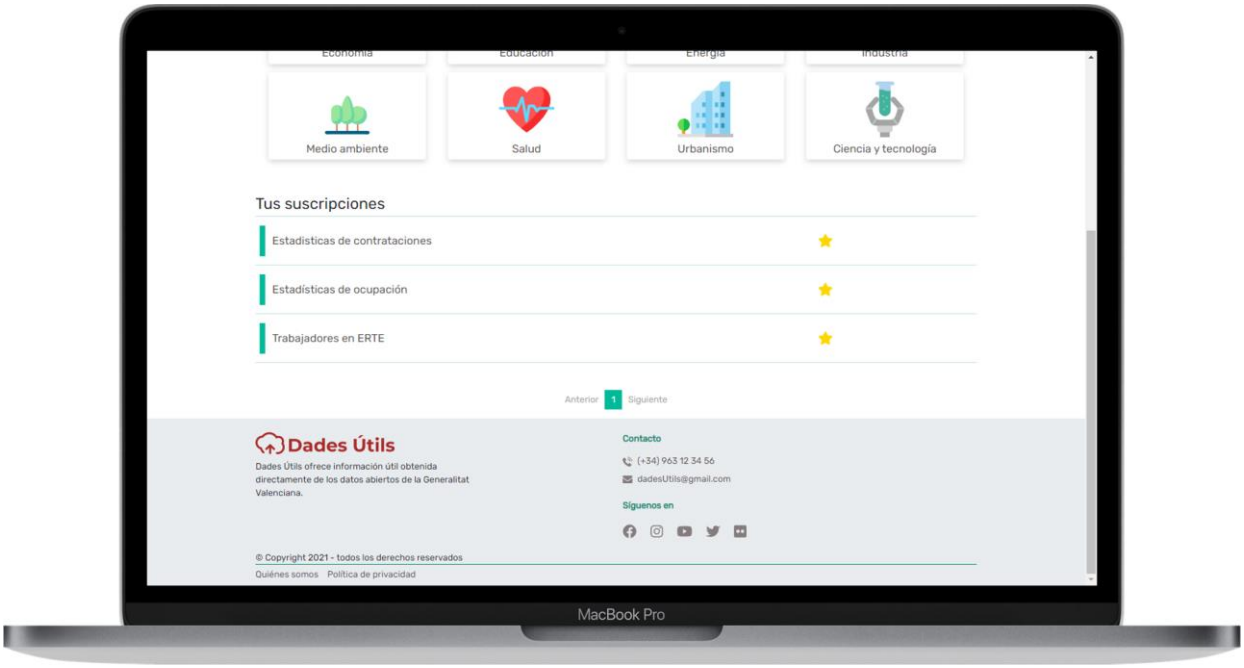

Figura 25: Captura de la parte inferior del menú principal, versión de ordenador.

Figura 26: Captura de la parte inferior del menú principal, versión de teléfono móvil.

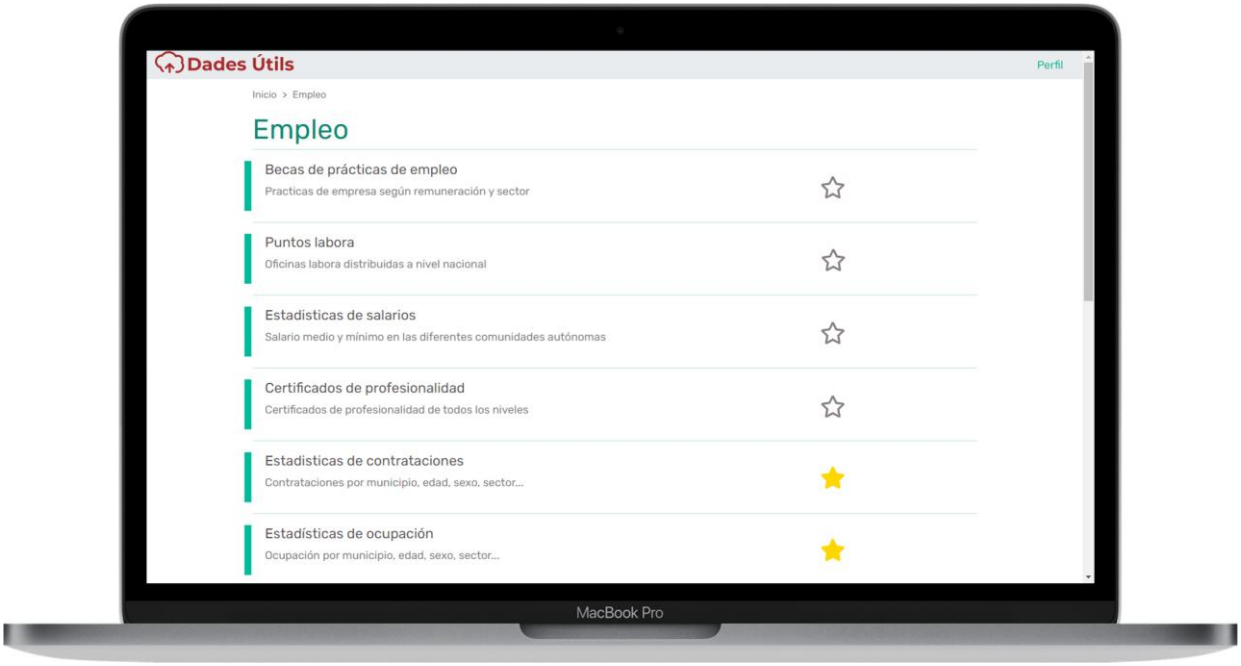

Figura 27: Captura de la lista de datasets de un tema, versión de ordenador.

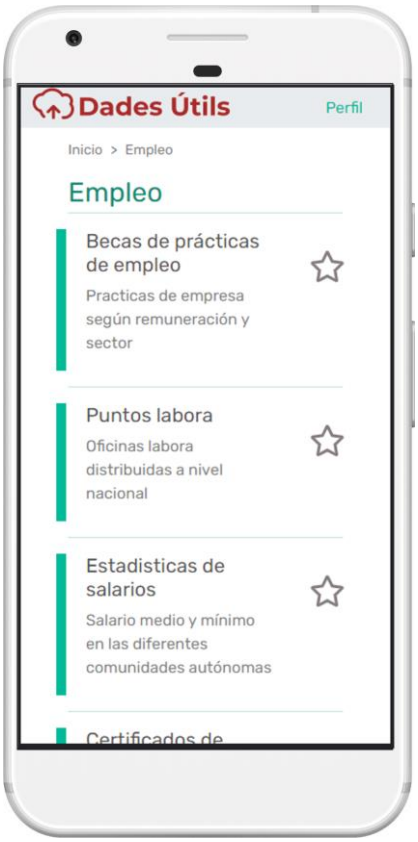

Figura 28: Captura de la lista de datasets de un tema, versión de teléfono móvil.

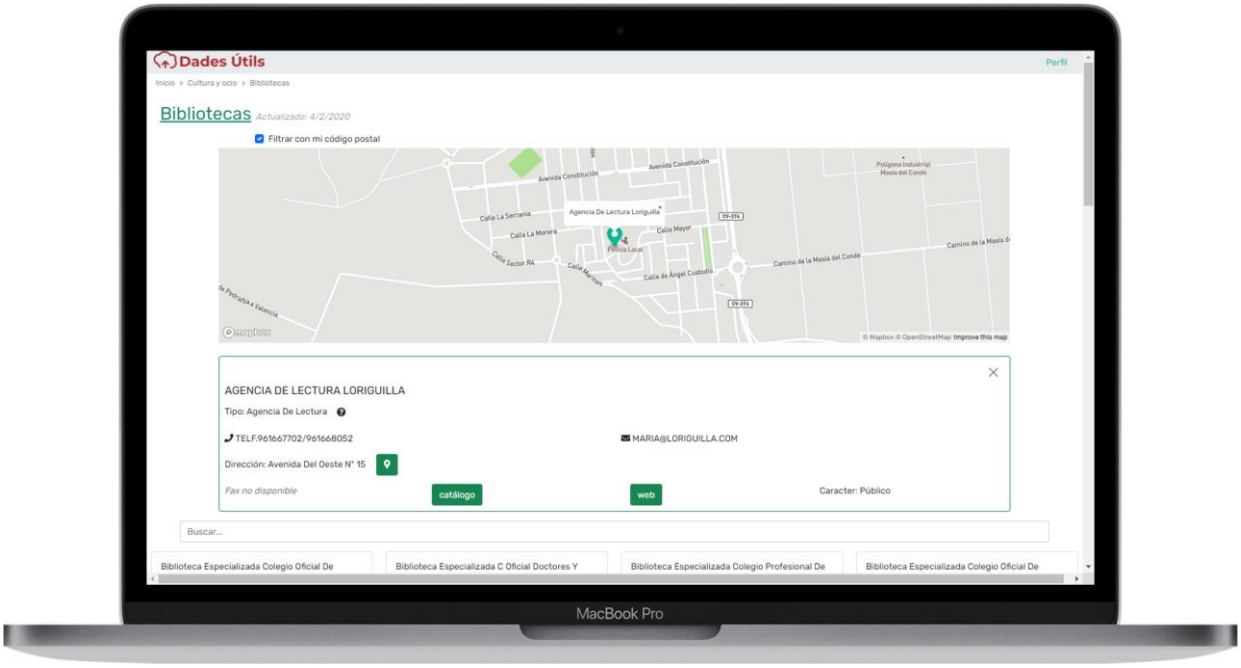

Figura 29: Captura de la vista de datos de "Bibliotecas", versión de ordenador.

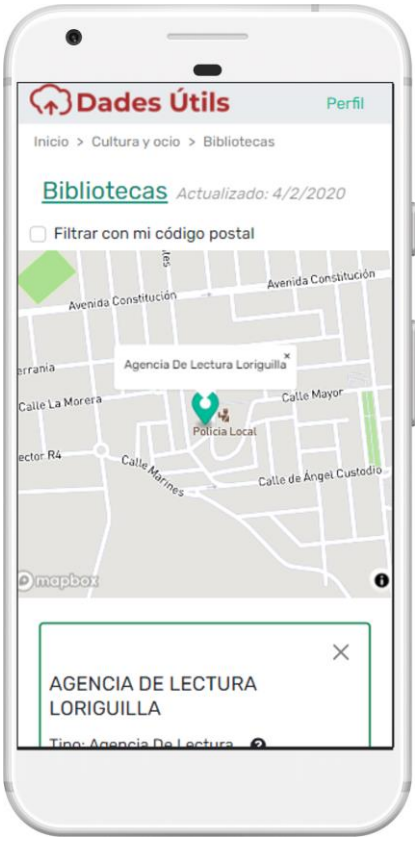

Figura 30: Captura de la vista de datos de "Bibliotecas", versión de teléfono móvil.

- Ofrecer datos de valor añadido, como medias y estadísticas, obtenidas mediante el procesamiento de los datos.

Este objetivo también se ha cumplido y un ejemplo evidente es el mapa que aparece en las figuras 27 y 28. Gracias a un sistema de Geocoding que traduce las direcciones en coordenadas es posible representar las bibliotecas en un mapa automáticamente.

- Hacer cómodo el acceso a datos que se quieran consultar regularmente.

Se ha implementado mediante la funcionalidad de suscripciones. Si el usuario pulsa en la estrella que hay al lado de cada dataset en las listas de las figuras 25 y 26, esos datasets se guardan como suscripciones. Cuando un dataset es una suscripción, aparecerá en varios sitios de la aplicación. Uno de estos sitios es la parte inferior del menú principal, se puede ver en las figuras 23 y 24. De esta forma quedan mucho más accesibles. Es posible eliminar una suscripción pulsando nuevamente en la estrella.

6.2 Conclusiones y relación con los estudios

Como se ha podido ver a lo largo de todo el trabajo, este proyecto ha tocado muchos de los aspectos del desarrollo software, tanto desde el punto de vista metodológico como del tecnológico. He llevado a cabo todas las fases del desarrollo yo mismo eligiendo las técnicas y tecnologías concretas a aplicar en cada momento. Y esto ha sido muy enriquecedor porque me ha dado la confianza de que en un futuro no dependeré de las

tecnologías que ya haya usado, podré ampliar mi repertorio por mi cuenta y escoger la que mejor se adapte a las características del proyecto. Creo que las tecnologías escogidas para este proyecto han sido acertadas porque no he percibido que me obstaculizasen en absoluto. De entre los beneficios que me ha proporcionado este proyecto podría empezar por lo más básico y general como que he aprendido varias tecnologías y he afianzado otras en las que solo tenía un conocimiento básico. Además las tecnologías utilizadas son punteras y muy populares, lo que es un buen aporte a la carrera laboral futura. Las practicas ágiles siempre me han parecido muy interesantes y beneficiosas pero en este proyecto además he experimentado el cómo afecta elegir unas u otras para aplicarlas y creo que ahora tengo un conocimiento más profesional, saber ver tanto las ventajas como los inconvenientes de las opciones disponibles para elegir la óptima.

Otra de las cosas que más valoro de este proyecto es que he podido comprobar que una preocupación que llevaba arrastrando los últimos años de la carrera es infundada. Al tocar en la carrera pocas tecnologías y algunas incluso desfasadas pensaba que cuando me tocara enfrentarme a las tecnologías punteras lo iba a tener muy difícil. Sin embargo me ha sorprendido gratamente ver que cuando tenía que recurrir a una librería, una herramienta o cualquier tecnología nueva solo tenía que ponerme a ello y acababa aprendiéndola rápidamente. Y esto es sin duda gracias a las bases que he adquirido en la carrera. Se me ha hecho evidente el valor que aportan asignaturas como programación, concurrencia y sistemas distribuidos, estructuras de datos y algoritmos, interfaces persona computador, lenguajes, teorías y paradigmas de la programación, redes de computadores, bases de datos y sistemas de la información etc. Y podría seguir con las de la rama, que me han aportado infinidad de conceptos sobre metodología, patrones de diseño, que aunque no los he implementado yo mismo en este proyecto sí que les he dado uso, patrones como el observador, interfaz etc.

Con respecto a las competencias transversales, creo que las que más he trabajado en este proyecto serían:

- Comprensión e integración: porque he tenido que comprender nuevas ideas y conceptos para incluirlos o no en el proyecto.
- Aplicación y pensamiento práctico: porque he tenido que tomar decisiones basadas en el conocimiento previo por ejemplo en la metodología
- Análisis y resolución de problemas: porque han surgido varios obstáculos grandes a lo largo del desarrollo que he tenido que examinar para encontrar la manera de resolverlos
- Pensamiento crítico: porque he tenido que juzgar las prácticas agiles pasando de un pensamiento de “todas son buenas, cuantas más mejor” a poder ver y analizar las ventajas e inconvenientes de cada una para el proyecto
- Instrumental específica: porque he tenido que aprender multitud de tecnologías nuevas.

7. Trabajos futuros

En esta sección se listan una serie de trabajos futuros que podrían llevarse a cabo para cumplir con los objetivos del proyecto aún más.

- Implementar las vistas de muchos más datasets con objetivo de que la página se popularice y con ella su demanda a las administraciones
- Mejorar la actualización automática de recursos
- Crear perfiles automatizados en redes sociales
- Solicitar la difusión de la página en el portal de datos abiertos de la GVA, pues tienen un formulario a tal efecto.
- Ocultar las API keys para poder hacer público el repositorio y así garantizar la no manipulación de los datos.

8. Referencias

[1]Organización Naciones Unidas "*Transformar nuestro mundo: la Agenda 2030 para el Desarrollo Sostenible*" septiembre 2015.

[2] «Principles | International Open Data Charter» opendatacharter.net/principles-es/ 2015. Consulta: 10-julio-2021.

[3] "*El Consell s'adhereix a la Carta Internacional de Dades Obertes i impulsa una estratègia per a implantar-la*" www.gva.es Mar 2021[4] "*I Pla biennal de transparència de la Generalitat 2019-2021*" www.gvaoberta.gva.es abril 2019. Consulta: 10-julio-2021.

[5] "*IV Plan de Gobierno Abierto 2020-2024*" transparencia.gob.es octubre 2020. Consulta: 13-julio-2021.

[6] "Catálogo de prácticas ágiles" agilev-roadmap.herokuapp.com/InfoPracticas. Consulta: 17-julio-2021.

[7] "Nine secretes of Kano model" Hienadz "Gena" Drahum slideshare.net/Hienadz.Drahun/understanding-kano-model 2014. Consulta: 20-julio-2021.

[8] "System Usability Scale (SUS)" www.usability.gov. Consulta: 5-agosto -2021.

[9] "Open data madurity report 2020" Laura van Knippenberg, data.europa.eu/sites/default/files/edp_landscaping_insight_report_n6_2020.pdf diciembre 2020. Consulta: 10-julio-2021.

[10] “Measuring open data madurity – Sixth Edition, 2020” European Data Portal (EDP) data.europa.eu/sites/default/files/method-paper_insights-report_n6_2020.pdf Consulta: 13-julio-2021.

[11] “Detailed country view” data.europa.eu Consulta: 15-julio-2021.

[12] “Análisis multidimensional de los portales de datos abiertos autonómicos españoles" Ricardo Curto-Rodríguez, redc.revistas.csic.es/index.php/redc/article/view/1314/2066 marzo 2021. Consulta: 22-julio-2021.

[13] “Anàlisi de l'ús del Portal de Dades Obertes de la Generalitat Valenciana en 2020” portaldadesobertes.gva.es/va/ Consulta: 23-julio-2021.

[14] “Constitución Española” www.boe.es, «BOE» núm. 311, de 29/12/1978. Consulta: 26-julio-2021.

[15] “Ley 19/2013, de 9 de diciembre, de transparencia, acceso a la información pública y buen gobierno” www.boe.es «BOE» núm. 295, de 10/12/2013. Consulta: 26-julio-2021.

[16] “Open Definition 2.1” opendefinition.org Consulta: 2-agosto-2021.

9. Anexo I: Detalle de la relación del proyecto

9.1 Relación con los principios de la Carta internacional de datos abiertos

Principio 2 - Oportunos y Exhaustivos

Este principio hace referencia, entre otras cosas, a que los datos tienen que ser relevantes. Para ello recomienda publicarlos de forma rápida y sin modificaciones. En el backlog de este trabajo se incluirá la función de actualizar automáticamente los datos presentados y procesarlos para, aparte de referenciar los originales, mostrar también información con valor añadido. De esta manera se vuelven más útiles para la ciudadanía y por ello más relevantes.

Principio 3 - Accesibles y Utilizables

Este principio hace referencia, entre otras cosas, a que los datos deben ser fácilmente legibles por máquinas y que se debe tener en cuenta la experiencia del usuario. Los datos presentados por la GVA cumplen con la primera parte, pero no con la segunda. Los datos son fácilmente

accesibles desde las API proporcionadas, pero para un ciudadano de a pie consultarlos directamente en el portal de la GVA es farragoso e ineficaz. En este trabajo se aprovechará el cumplimiento del primer aspecto para proporcionar el segundo.

9.2 Relación con los puntos del Plan bienal de transparencia 2019-2021

Línea estratégica 1 – Garantizar el acceso efectivo a la información pública

Punto 32. Mejora del diseño y visualización de los contenidos del portal GVA Oberta

En este trabajo se procurará plantear los datos de una forma fácil de visualizar. Ayudando a que la ciudadanía pueda consumirlos más cómodamente, repercutiendo así en la eficacia de la publicación de los datos.

Línea estratégica 2 – Promover la cultura de la transparencia entre la ciudadanía

Punto 34. Estrategia de comunicación y difusión del gobierno abierto

En este trabajo se incluirán las referencias y los enlaces a las fuentes de datos. Ayudando así a dar a conocer el portal de datos abiertos de la GVA. Este punto es crucial pues por mucha inversión y esfuerzos que se hagan de cara a tener un buen portal de transparencia y datos abiertos, de nada servirá si la población no lo conoce.

Línea estratégica 4 – Crear un marco de planificación, evaluación y redición de cuentas de las políticas públicas

Punto 57. Mejora de la rendición de cuentas de la Generalitat

Una parte importante de la rendición de cuentas es la publicación de los datos en los que se basan las medidas políticas. Esto queda recogido también en la Carta internacional de datos abiertos mencionada anteriormente.

10. Anexo II: Relación con los ODS

OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	**Alto**	**Medio**	**Bajo**	**No Procede**
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**				X
ODS 4. **Educación de calidad.**				X
ODS 5. **Igualdad de género.**				X
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**			X	
ODS 9. **Industria, innovación e infraestructuras.**				X
ODS 10. **Reducción de las desigualdades**.				X
ODS 11. **Ciudades y comunidades sostenibles**.				X
ODS 12. **Producción y consumo responsables.**				X
ODS 13. **Acción por el clima.**				X
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**	X			
ODS 17. **Alianzas para lograr objetivos.**				X

Reflexión sobre la relación del TFG con los ODS y con el ODS más relacionado.

En un portal como el desarrollado en este proyecto podría llegar a haber datos muy variados. Es por ello por lo que podría llegar a estar relacionado con muchos ODS en función de las temáticas de los datos. Sin embargo, el principal objetivo con el que está estrechamente relacionado es el 16 “Paz, justicia e instituciones sólidas”. A continuación se explica la relación con las metas en concreto.

> Meta 16.6 - Crear a todos los niveles instituciones eficaces y transparentes que rindan cuentas

Los datos que se recogen por parte de las administraciones normalmente van destinados a la elaboración de medidas legislativas y políticas. También, en declaraciones públicas y en la motivación y objetivos de las iniciativas legislativas se utilizan estos datos. Una parte básica de la transparencia tiene que poder acceder a los datos en los que se han basado conclusiones que han acabado fundamentando leyes y ordenanzas de todo tipo. Presentando los datos de una manera fácil de consumir se facilita el entender las bases de las decisiones políticas que se justifican con estos datos.

Meta 16.7 - Garantizar la adopción en todos los niveles de decisiones inclusivas, participativas y representativas que respondan a las necesidades

Para que la ciudadanía pueda juzgar adecuadamente si las decisiones tomadas a partir de datos son o no representativas de su voluntad es necesario que esa ciudadanía pueda consumirlos y crear conocimiento a partir de la información que se ha recabado. Si la ciudadanía no tiene los medios para dar un “feed back” a la clase política no puede asegurarse la representatividad. Mediante la presentación de los datos, este trabajo facilita a cada ciudadano el formarse un criterio fundado al respecto.

Meta 16.10 - Garantizar el acceso público a la información y proteger las libertades fundamentales, de conformidad con las leyes nacionales y los acuerdos internacionales

Si bien es cierto que podría interpretarse que habilitando unas API y una web donde se pueden visualizar varios conjuntos de datos en bruto ya se está garantizando el acceso a la información, creo que desde un punto de vista práctico la medida no es suficiente y no cumple con el espíritu de la meta. Es necesario que estos datos se presenten de una manera fácil de consumir y de obtener conocimiento a partir de ellos. Lo que se va a acometer en este trabajo debería ser lo que hicieran las administraciones públicas que tienen datos recabados.

11. Glosario de términos y acrónimos

A continuación se definen algunos términos y acrónimos empleados a lo largo del trabajo.

- Backlog: Sinónimo en este proyecto de Product Backlog. Listado de tareas que hay que realizar para el desarrollo del producto. Incluidas implementaciones y trabajos necesarios para desarrollos como estudiar una API.
- Conjunto de datos: Agrupación de recursos por temática

- Dades Útils: Nombre de la aplicación web que se desarrolla a lo largo del proyecto
- Dataset: Traducción al inglés de “Conjunto de datos”.
- Dato: Unidad mínima de información recabada por las administraciones públicas.
- Datos en bruto: Datos sin ningún procesamiento.
- Entregas: Despliegues al entorno de producción.
- Ítems: Cada uno de los elementos que componen el backlog.
- MVP: Acrónimo para Most Valuable Player, se utiliza en metodología software para identificar al item del backlog de mayor prioridad.
- Plataforma de despliegue: En este caso Heroku, la plataforma que mantiene el despliegue en producción y actúa como servidor permanente para servir las peticiones.
- Recurso: Agrupa los datos por temática, fecha etc.
- SEO: Acrónimo para “Search Engine Optimization”. Es el conjunto de técnicas que hacen más fácil para el motor de búsqueda el leer la página y afecta a la facilidad para que salga de los primeros resultados en las búsquedas.
- Tareas: Sinónimo de ítems. Cada uno de los elementos que componen el Backlog.
- Unidades de trabajo: sinónimo de tareas o ítems. Cada uno de los elementos que componen el backlog