

*PROYECTO:*

# **DATOS ABIERTOS DEMOGRAFICOS JUNTA CYL**



*Realizado por:*

**NOMBRE EQUIPO: DEMOGRAFICOS MUNICIPALES.**

- *Samuel Fernández*
- *Álvaro García*
- *Luca Mascani*



## Desarrollo de Aplicaciones Web (DAW)

Valladolid, a 5 de Febrero de 2026

### Contenido

1.	Introducción .....	2
2.	Análisis.....	3
2.1.	<i>Dataset</i> utilizado .....	3
2.2.	Requisitos funcionales .....	4
2.3.	Diagramas de casos de uso .....	5
3.	Diseño.....	6
3.1.	Modelo de base de datos.....	6
3.2.	Diseño de la interfaz .....	8
4.	Desarrollo .....	12
4.1.	<i>Stack</i> tecnológico .....	12
4.2.	Estructura de carpetas .....	13
4.3.	Descripción del funcionamiento .....	14
5.	Pruebas.....	16
6.	Despliegue .....	18
7.	Sostenibilidad y "Green Coding" .....	21
8.	Conclusiones.....	22
8.1.	Autoevaluación .....	22
8.2.	Líneas futuras.....	23
9.	Bibliografía.....	23

## 1. Introducción

Hoy en día tenemos acceso a muchísimos datos públicos, pero de nada sirven si no podemos visualizarlos de forma clara. Este proyecto, Demografía CyL, nace con un objetivo práctico: transformar los datos "en crudo" de la Junta de Castilla y León en información útil y fácil de entender para cualquier persona.

La aplicación se centra en analizar el Movimiento Natural de la Población (nacimientos, defunciones y matrimonios) en nuestra comunidad. En lugar de descargar hojas de cálculo o leer tablas interminables, el sistema se conecta directamente a las APIs oficiales de la Junta, procesa la información histórica (2020-2023) y la presenta mediante gráficos interactivos y mapas claros.

Técnicamente, es una solución web completa desarrollada con Laravel 11 y Tailwind CSS. El reto principal ha sido crear un sistema capaz de gestionar y actualizar la información de los más de 2.200 municipios que tiene Castilla y León, ofreciendo una experiencia de usuario fluida, moderna y visualmente atractiva.



Figura 1.Logo de la pagina.

Tabla 1. Ficha técnica del proyecto DatosJuntaCyL.

Nombre del proyecto	Tipo de software	Fuente de datos	Cobertura geográfica	Periodo analizado	Tecnologías principales
DatosJuntaCyL (Demografía CyL)	Aplicación Web de Visualización de Datos (Dashboard)	Portal de Datos Abiertos de la Junta de Castilla y León	9 provincias y +2.200 municipios de CyL	En nuestro proyecto: Series históricas del MNP (2020 - 2023)	Laravel 11, Tailwind CSS 4.0, Chart.js, MySQL

## 2. Análisis

### 2.1. Dataset utilizado

- **Nombre:** Consultas al movimiento natural de la población, registro de municipios de castilla y león , y municipios de castilla y león

- **URL:**

[https://datosabiertos.jcyl.es/web/jcyl/set/es/demografia/movimiento\\_poblacion/1284208120307](https://datosabiertos.jcyl.es/web/jcyl/set/es/demografia/movimiento_poblacion/1284208120307)

<https://datosabiertos.jcyl.es/web/jcyl/set/es/sector-publico/municipios/1284278782067>

<https://analisis.datosabiertos.jcyl.es/api/explore/v2.1/catalog/datasets/registro-de-municipios-de-castilla-y-leon/records>

- **Campos utilizados:**

- *Tabla 2. Ficha técnica de campos utilizados de la api .*

Fuente de Datos	Campo API (Origen)	Campo Base de Datos (Destino)	Notas / Transformación
<b>1. API MNP (Movimiento Natural)</b>	COD_MUNICIPIO	municipio_id	Foreign Key
	ANNO	anno	
	VALOR_VARIABLE	valor	
	COD_FAMILIA_VARIABLES	tipo_evento	<b>Mapeo de valores:</b> 10 → 'nacimiento' 20 → 'matrimonio' 30 → 'defuncion'
<b>2. API OpenDataSoft (Registro Municipios)</b>	Cod_INE	municipios.codigo_ine	
	Municipio	municipios.nombre	
	Cod_Provincia	provincias.codigo_ine	
	Provincia	provincias.nombre	
<b>3. API OpenDataSoft (Población)</b>	cod_ine	municipios.codigo_ine	<b>Clave de búsqueda</b>
	poblacion	municipios.poblacion	

## 2.2. Requisitos funcionales

El sistema implementa las siguientes funcionalidades divididas por módulos:

### Módulo de Visualización y Análisis

- RF-01. Dashboard General: Visualización de tarjetas con contadores globales (total de provincias, municipios y registros), resumen de eventos MNP y tablas de clasificación ("Top 10 Municipios más activos").
- RF-02. Fichas de Detalle: Generación de vistas detalladas por provincia y municipio que incluyen gráficos de evolución temporal (líneas), distribución de eventos y tablas de registros históricos.
- RF-03. Comparador Interactivo: Herramienta que permite seleccionar dos provincias distintas y visualizar una comparativa "lado a lado" de sus métricas, calculando automáticamente la diferencia numérica entre ambas.
- RF-04. Mapas y Gráficos: Representación visual de datos mediante gráficos interactivos (Chart.js) y acceso a mapas de calor para identificar densidades demográficas.

### Módulo de Navegación y Búsqueda

- RF-05. Buscador y Filtros: Sistema de búsqueda en tiempo real para localizar provincias o municipios por nombre y código INE.

### Módulo de Usuario y Personalización

- RF-06. Gestión de Perfil: Área privada donde el usuario puede consultar su información y acceder a configuraciones personales.

- RF-07. Sistema de Favoritos: Funcionalidad que permite marcar municipios específicos como "Favoritos" para acceder rápidamente a ellos desde un panel dedicado y realizar un seguimiento personalizado
- RF-08. Autenticación Segura: Sistema de registro e inicio de sesión de usuarios protegido mediante CAPTCHA para prevenir accesos automatizados y ataques de fuerza bruta.

### 2.3. Diagramas de casos de uso

Ejemplo: <https://drawio-app.com/blog/uml-use-case-diagrams-with-draw-io/>

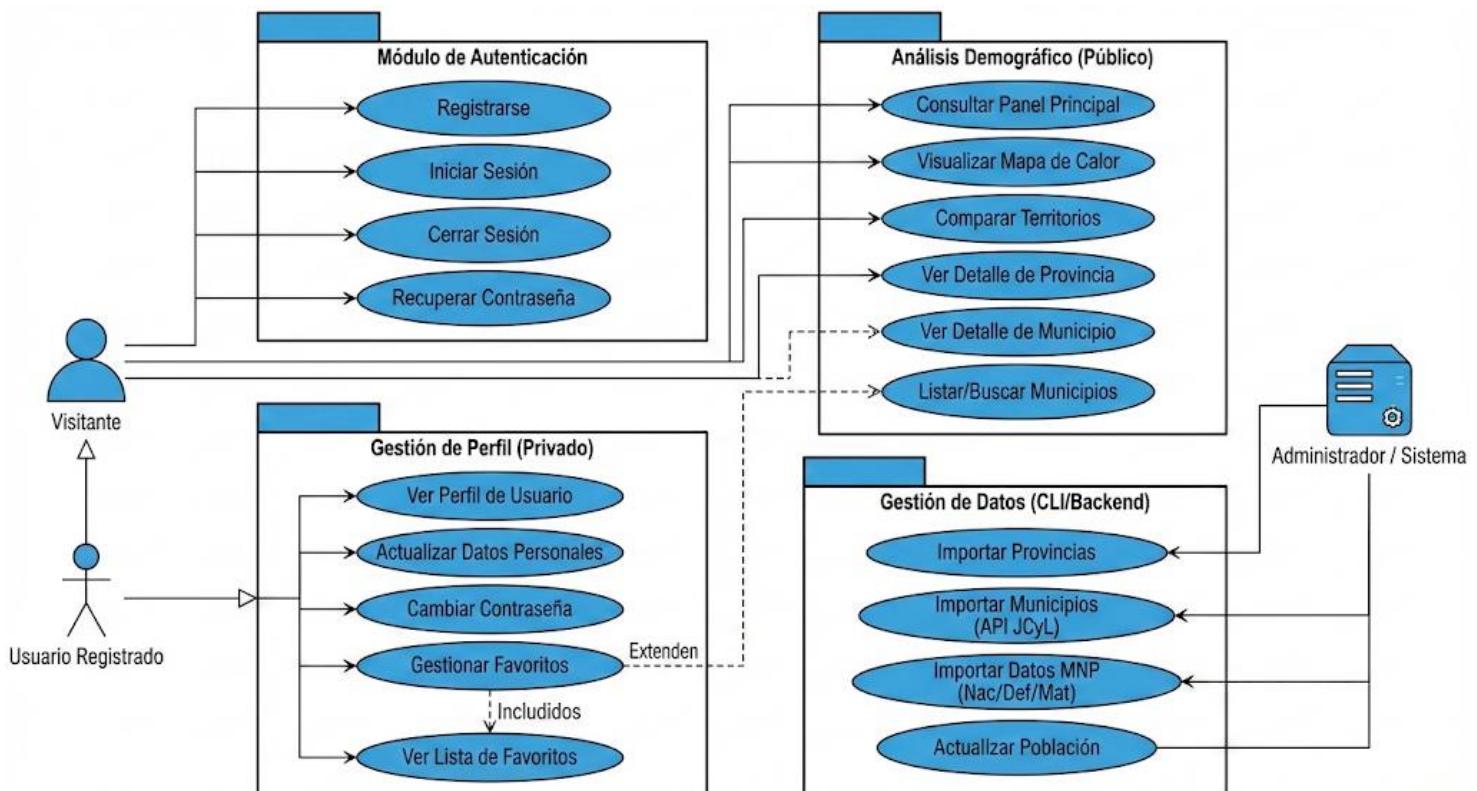


Figura 2. Diagrama de casos de uso

## 3. Diseño

### 3.1. Modelo de base de datos

- Diagrama Entidad-Relación:

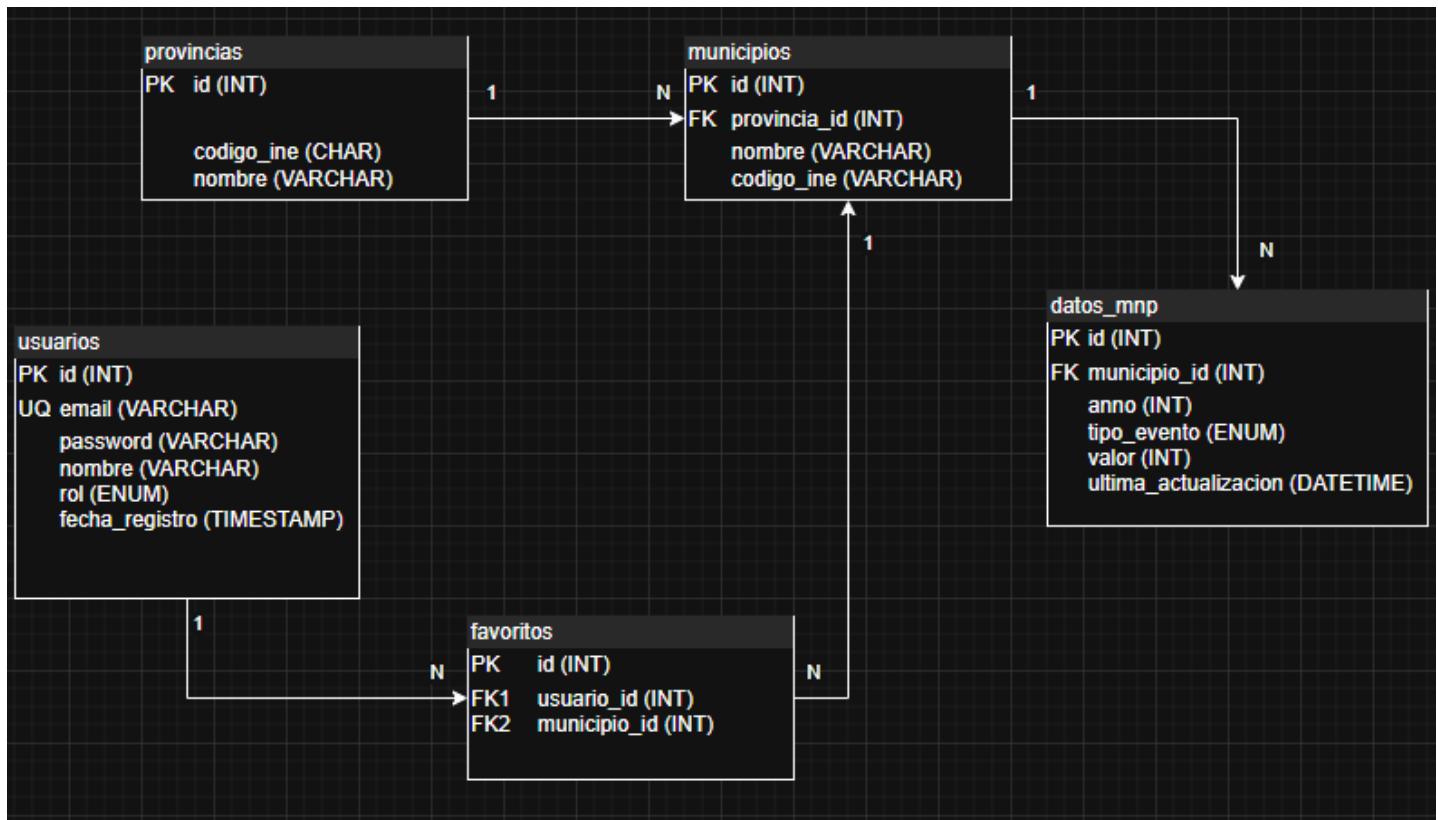


Figura 3. Diagrama entidad Relación

### 1. Normalización Geográfica (Relación 1:N)

Se han separado las entidades provincias y municipios para cumplir con la Tercera Forma Normal (3NF).

- **Relación:** Una provincia tiene muchos municipios (1:N).
- **Justificación:** Almacenar el nombre de la provincia en cada registro de municipio generaría redundancia excesiva. Al separarlas, optimizamos el almacenamiento y facilitamos la gestión; si fuera necesario corregir el nombre de una provincia, se realiza en un único lugar y se propaga automáticamente.

## **2. Diseño de Series Temporales (Tabla datos\_mnp)**

En lugar de crear columnas por años (ej. nacimientos\_2020, nacimientos\_2021), se ha optado por un diseño vertical mediante la tabla datos\_mnp.

- **Relación:** Un municipio tiene múltiples registros de datos demográficos (1:N).
- **Justificación (Escalabilidad):** Este diseño permite que el sistema sea escalable infinitamente en el tiempo. Cuando la Junta de Castilla y León publique los datos de 2024 o 2025, no será necesario modificar la estructura de la base de datos (añadir columnas), simplemente se insertarán nuevas filas. Además, facilita enormemente las consultas de agregación SQL (ej. SUM(valor) WHERE anno BETWEEN 2020 AND 2023).

## **3. Personalización y Favoritos (Relación N:M)**

Para la funcionalidad de "Municipios Favoritos", se ha modelado una relación de **Muchos a Muchos** entre usuarios y municipios.

- **Implementación:** Se utiliza una tabla pivot o intermedia llamada favoritos.
- **Justificación:** Un usuario puede seguir a varios municipios y un municipio puede ser seguido por múltiples usuarios. La tabla intermedia almacena únicamente las claves foráneas (usuario\_id, municipio\_id), lo que permite consultas indexadas muy rápidas y evita la creación de campos complejos dentro de la tabla de usuarios.

## **4. Integridad y Rendimiento**

- **Claves Primarias:** Se utilizan identificadores numéricos (id autoincrementales) para todas las relaciones internas (Foreign Keys), dejando los códigos INE (codigo\_ine) como campos indexados para búsquedas y cruce de datos con la API externa. Esto mejora el rendimiento de los JOINs comparado con el uso de cadenas de texto.
- **Integridad Referencial:** El modelo asegura que no puedan existir datos demográficos huérfanos (sin municipio asociado) ni favoritos de usuarios inexistentes.

### 3.2. Diseño de la interfaz

- Prototipo del *layout* principal para las vistas *desktop* y *mobile*.

The screenshot shows the main landing page of the demographic data portal. At the top, there is a header bar with a logo, navigation links for 'Panel', 'Comparar', 'Provincias', 'Municipios', and 'Mapa de Calor', and a user profile section for 'SAMUEL'. Below the header, the title 'Portal de Datos Demográficos' is prominently displayed in large red font. A subtitle 'Análisis integral del Movimiento Natural de la Población de Castilla y León' follows. Two red buttons labeled 'Ver Mapa →' and 'Ver Datos →' are visible. The main content area is titled 'Acceso Rápido' and contains three cards: 'Panel Principal' (with a chart icon), 'Provincias' (with a map icon), and 'Municipios' (with a house icon). Each card has a brief description and a red 'Ver [Category]' button. Below these cards are two more cards partially visible: one for 'Movimiento Natural' (with a person icon) and another for 'Mapa de Calor' (with a sun icon).

Figura 4. Pagina principal(vista ordenador)



Figura 5. Pagina principal(vista móvil)

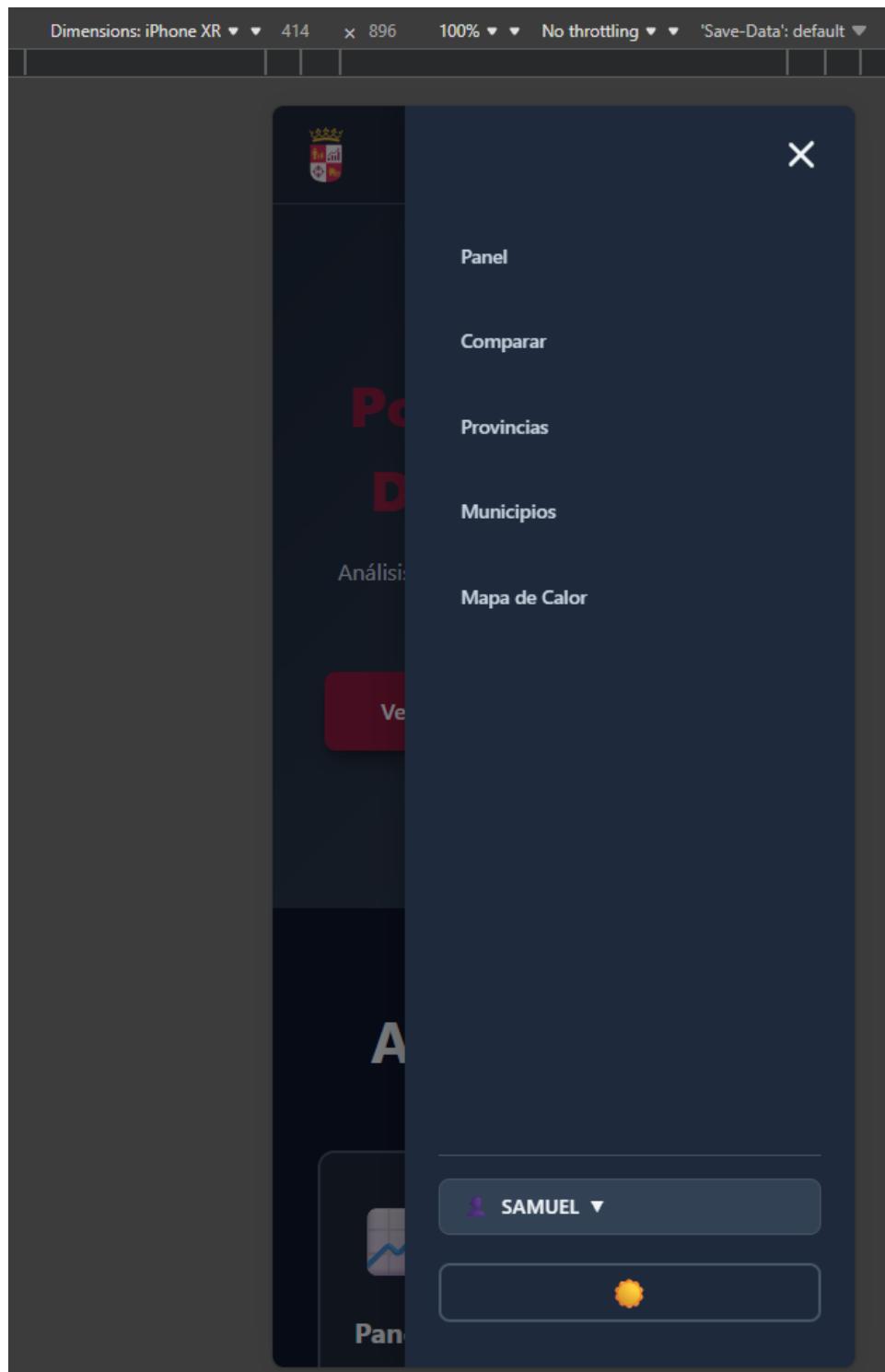


Figura 6. Pagina principal(menú en vista móvil)

- **Colores:**

La identidad visual del proyecto se basa en el color corporativo "Granate Castilla", complementado por escalas neutras de grises azulados (Slate) para la interfaz moderna.

**1. Colores Primarios (Identidad Corporativa)** Se utiliza una tríada monocromática basada en el rojo purpúreo característico de la región:

- **Primary Base:** #8b1a3f (Usado en cabeceras, botones principales y elementos activos).
- **Primary Light:** #a8324a (Estados *hover* y degradados).
- **Primary Dark:** #6b1430 (Bordes y contrastes altos).

**2. Colores Neutros y Modo Oscuro** El sistema implementa variables CSS (:root vs [data-tema="oscuro"]) para adaptar la interfaz dinámicamente:

- **Modo Claro:** Fondos en blanco puro (#ffffff) y grises muy claros (#f9fafb) con texto en gris oscuro (#1f2937) para maximizar el contraste de lectura.
- **Modo Oscuro (Ahorro Energético):** Se emplean tonos profundos de la paleta *Slate* (#0f172a para fondo principal y #1e293b para tarjetas). Estos colores oscuros reducen el consumo en pantallas OLED y disminuyen la fatiga visual.

**3. Colores Semánticos (Feedback)** Se utilizan colores funcionales estándar para comunicar estados del sistema al usuario:

- **Éxito (Verde):** #10b981 (Ej. "Guardado en favoritos").
- **Advertencia (Ámbar):** #f59e0b (Ej. Alertas de datos no disponibles).
- **Error (Rojo):** #ef4444 (Ej. Fallo de validación en login).
- **Tipografía:**

Se ha optado por una pila de fuentes (font-stack) del sistema moderna ("System UI") encabezada por **Instrument Sans**, garantizando que la aplicación se renderice con la tipografía nativa más optimizada para cada sistema operativo sin necesidad de descargar fuentes pesadas innecesariamente.

- **Familia Tipográfica:** Instrument Sans, system-ui, -apple-system, Segoe UI, Roboto, Helvetica Neue, sans-serif.
- **Pesos utilizados:**
  - *Regular (400)*: Cuerpo de texto y párrafos.
  - *Medium (500)*: Enlaces de navegación y etiquetas de formulario.
  - *Semibold (600) / Bold (700)*: Títulos de tarjetas, encabezados de página (h1, h2) y cifras destacadas en el dashboard.

## 4. Desarrollo

### 4.1. *Stack tecnológico*

#### Entorno: Backend (Servidor)

- **Lenguaje:** PHP 8.2. Se ha elegido esta versión por su mejoras en el sistema de tipos y rendimiento.
- **Framework:** **Laravel 11.** Utilizado como núcleo del proyecto para la gestión de rutas, controladores y la interacción con la base de datos a través del ORM Eloquent.
  - *Librerías destacadas:* GuzzleHTTP (para el consumo de las APIs de la Junta) y Artisan (para la creación de comandos personalizados de importación de datos).
- **Base de Datos:** **MySQL 8.0.** Sistema de gestión de bases de datos relacional donde se almacena la información de los más de 2.200 municipios y el histórico del MNP.

#### Frontend (Cliente)

- **Motor de Plantillas:** **Blade.** Integrado nativamente en Laravel para la generación de vistas dinámicas.
- **Estilos:** **Tailwind CSS 4.0.** Framework de utilidad para un diseño rápido, moderno y totalmente responsive (Mobile First).
- **Scripting:** **JavaScript (ES6).** Utilizado para la lógica del lado del cliente, peticiones asíncronas (AJAX/Fetch) y la interactividad de la interfaz.
- **Visualización de Datos:** **Chart.js.** Librería empleada para la renderización de los gráficos de barras, líneas y donuts que muestran las estadísticas demográficas.
- **Empaquetador:** **Vite 7.0.** Herramienta de compilación que optimiza la carga de recursos (assets) y permite la recarga en caliente durante el desarrollo.
- **Software:**
  - Visual Studio Code
  - Draw.io
  - Laragon
  - Git / Github
  - RailWay

#### 4.2. Estructura de carpetas

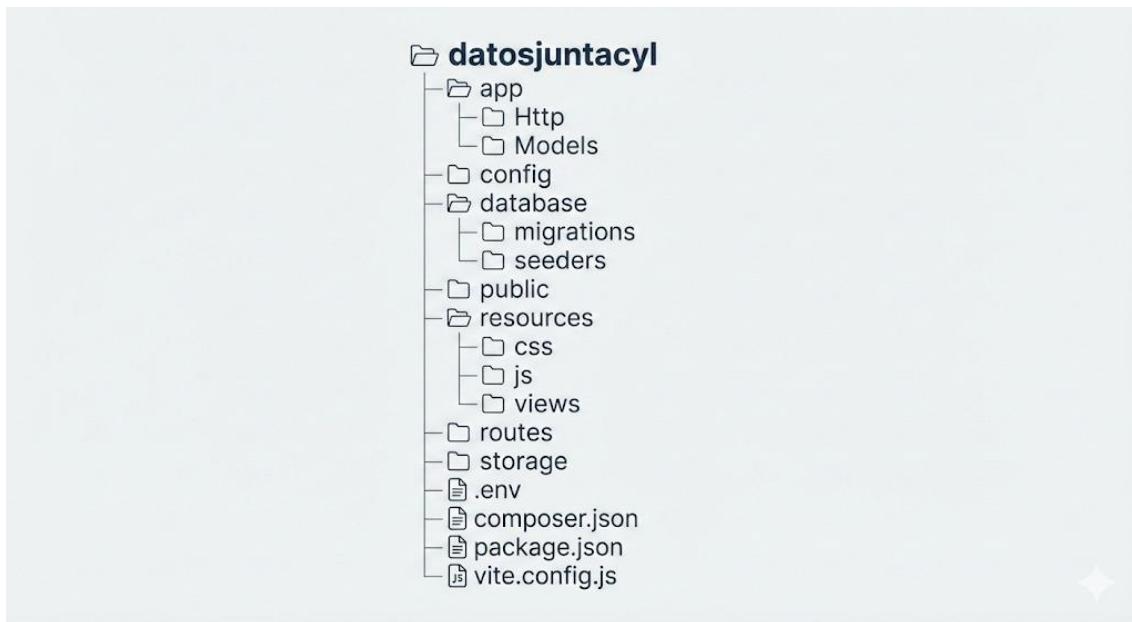


Figura 7. Estructura de carpetas

**app/**: Contiene la lógica de negocio principal de la aplicación.

- **Console/Commands/**: Comandos personalizados de Artisan creados para la ingestión de datos (ej. ImportMnpData.php, ActualizarPoblacionMunicipio.php).
- **Http/Controllers/**: Controladores que gestionan las peticiones web (ej. ControladorAnalisisDemografico.php, MunicipioController.php).
- **Models/**: Modelos Eloquent que representan las tablas de la base de datos (ej. DatoMnp.php, Municipio.php).
- **Services/**: Lógica externa separada, como Mnp ApiService.php para la conexión con la API de la Junta.

**config/**: Archivos de configuración global del framework (base de datos, caché, servicios, etc.).

**database/**: Gestión de la base de datos.

- **migrations/**: Archivos para crear y modificar la estructura de las tablas (ej. creación de tablas datos\_mnp, municipios).
- **seeders/**: Clases para poblar la base de datos con información inicial (ej. ProvinciaSeeder, UsuarioSeeder).

**public/**: Directorio público accesible desde el navegador.

- **geojson/**: Archivos JSON con las geometrías para los mapas interactivos.
- **img/**: Recursos gráficos como logotipos (LOGOTEXTO.png).

- **build/**: Archivos CSS y JS compilados y minificados por Vite para producción.

**resources/**: Recursos sin compilar y vistas.

- **css/ y js/**: Archivos fuente de Tailwind CSS y JavaScript (ej. app.js, favoritos.js).
- **views/**: Plantillas Blade para el frontend (organizadas en carpetas como análisis-demográfico, auth, componentes).

**routes/**: Definición de las rutas de la aplicación (web.php para las rutas del navegador y console.php para comandos).

**storage/**: Almacenamiento de archivos generados por Laravel, como logs, caché de vistas y sesiones.

**tests/**: Pruebas automatizadas (Unitarias y de Feature) para asegurar la calidad del código.

**vendor/**: Dependencias de PHP instaladas vía Composer (no se sube al repositorio).

#### Archivos raíz:

- **.env**: Variables de entorno (credenciales de BD, configuración local).
- **compose.json**: Definición de dependencias de PHP.
- **package.json** y **vite.config.js**: Configuración de dependencias de Node.js y del empaquetador Vite.

### 4.3. Descripción del funcionamiento

#### 1. Página de Inicio y Navegación Pública

Al acceder a la aplicación, el usuario es recibido por una página informativa que presenta el propósito del proyecto y un resumen visual del alcance de los datos (años disponibles y tipos de eventos demográficos).

- **Barra de Navegación**: Situada en la parte superior, es persistente en toda la aplicación y permite el acceso directo a las secciones principales: *Inicio*, *Panel*, *Provincias*, *Municipios*, *Mapa de calor* y el acceso al área de *Usuario*.

#### 2. Módulo de Autenticación

Aunque la consulta general de datos es pública, para disfrutar de una experiencia personalizada es necesario disponer de una cuenta.

- **Registro**: El usuario puede crear una cuenta proporcionando un nombre, correo electrónico y contraseña. El formulario incluye validación en tiempo real de los campos.

- **Inicio de Sesión:** Permite el acceso a usuarios registrados. Una vez autenticado, el menú de navegación cambia para mostrar el avatar del usuario y acceso a su *Perfil* y *Favoritos*.

### **3. Panel de Análisis Demográfico (Dashboard)**

Es el núcleo de la aplicación. Al entrar en la sección "Análisis", el usuario visualiza un **Dashboard Interactivo** que carga los datos de forma asíncrona para no bloquear la interfaz.

- **Visualización de Datos:**
  - **Tarjetas de Resumen:** Muestran los totales absolutos del periodo seleccionado.
  - **Top Municipios:** Una tabla clasificada muestra los 10 municipios con mayor actividad para la métrica seleccionada.

### **4. Fichas de Detalle**

Al hacer clic en una provincia o municipio desde cualquier listado o mapa, se accede a su **Ficha Detallada**.

- **Información Contextual:** Se muestra el nombre, código INE y población actual del municipio (si esta disponible).
- **Histórico Completo:** Se despliegan gráficos específicos que comparan los nacimientos frente a las defunciones (Crecimiento Vegetativo) a lo largo de los años disponibles en el sistema.

### **5. Herramienta de Comparación**

Esta funcionalidad permite seleccionar dos entidades territoriales (dos provincias) para visualizarlas "lado a lado".

- El sistema genera una vista dividida donde se contrastan sus métricas clave.
- Se calcula automáticamente el **diferencial numérico** y porcentual entre ambas entidades, facilitando el análisis de disparidades demográficas entre zonas rurales y urbanas.

### **6. Mapas de Calor**

En la sección de mapas, la aplicación renderiza un mapa interactivo de Castilla y León utilizando datos GeoJSON.

- Las provincias o municipios se colorean en una escala de intensidad (Heatmap) dependiendo del volumen de datos (ej. mayor número de nacimientos = color más intenso), permitiendo identificar rápidamente los focos de actividad demográfica en la comunidad.

## 7. Área Personal y Favoritos

Los usuarios registrados disponen de un panel de control propio:

- **Gestión de Favoritos:** En cualquier ficha de municipio, el usuario puede pulsar el icono de "Corazón". Esto añade el municipio a su lista de seguimiento rápido en el perfil.
- **Acceso Rápido:** Desde el perfil, el usuario puede ver una lista de sus municipios de interés y saltar directamente a sus estadísticas sin tener que buscarlos nuevamente.

## 5. Pruebas

### Prueba de accesibilidad con wave

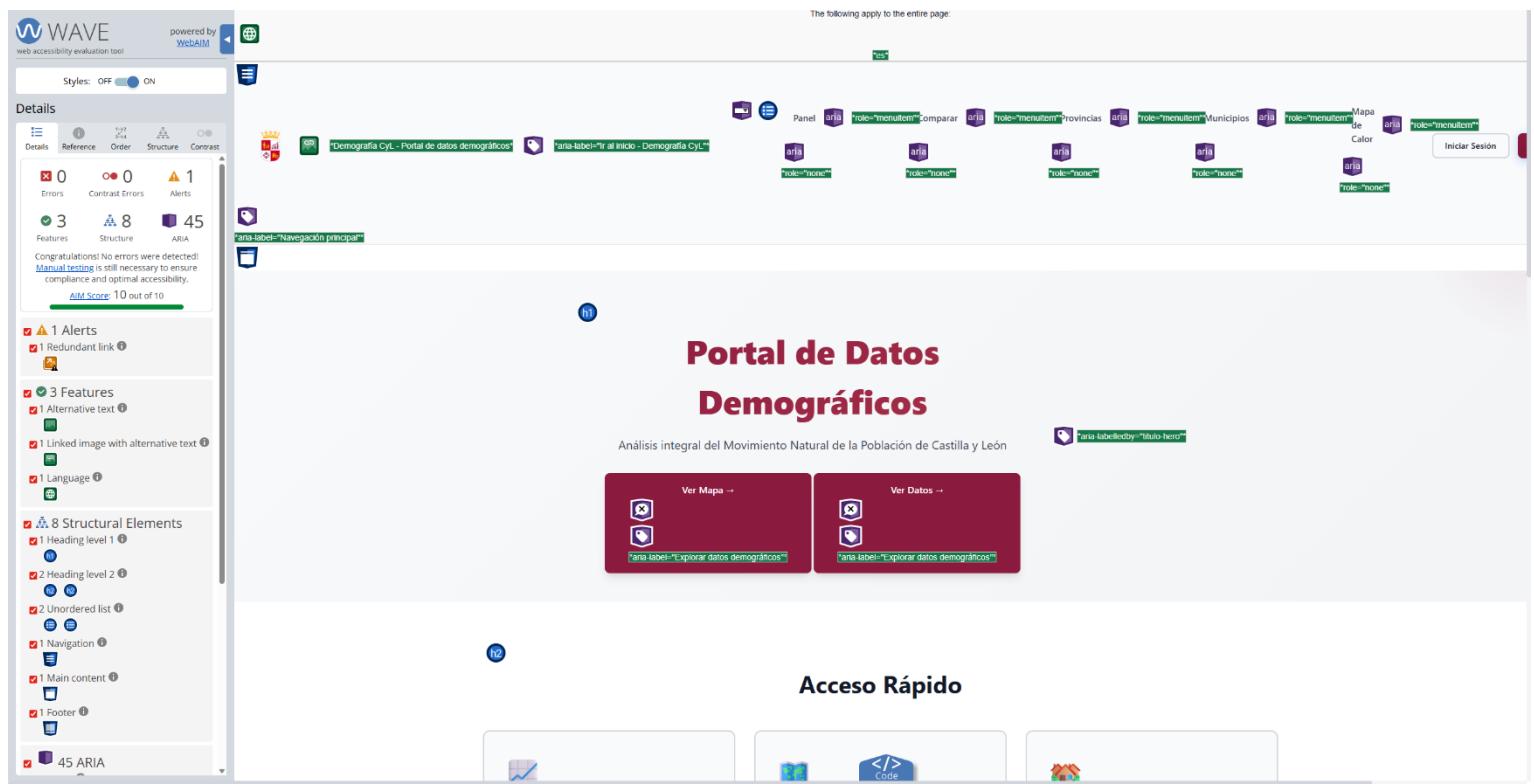


Figura 8. Prueba de accesibilidad con wave (pagina principal)

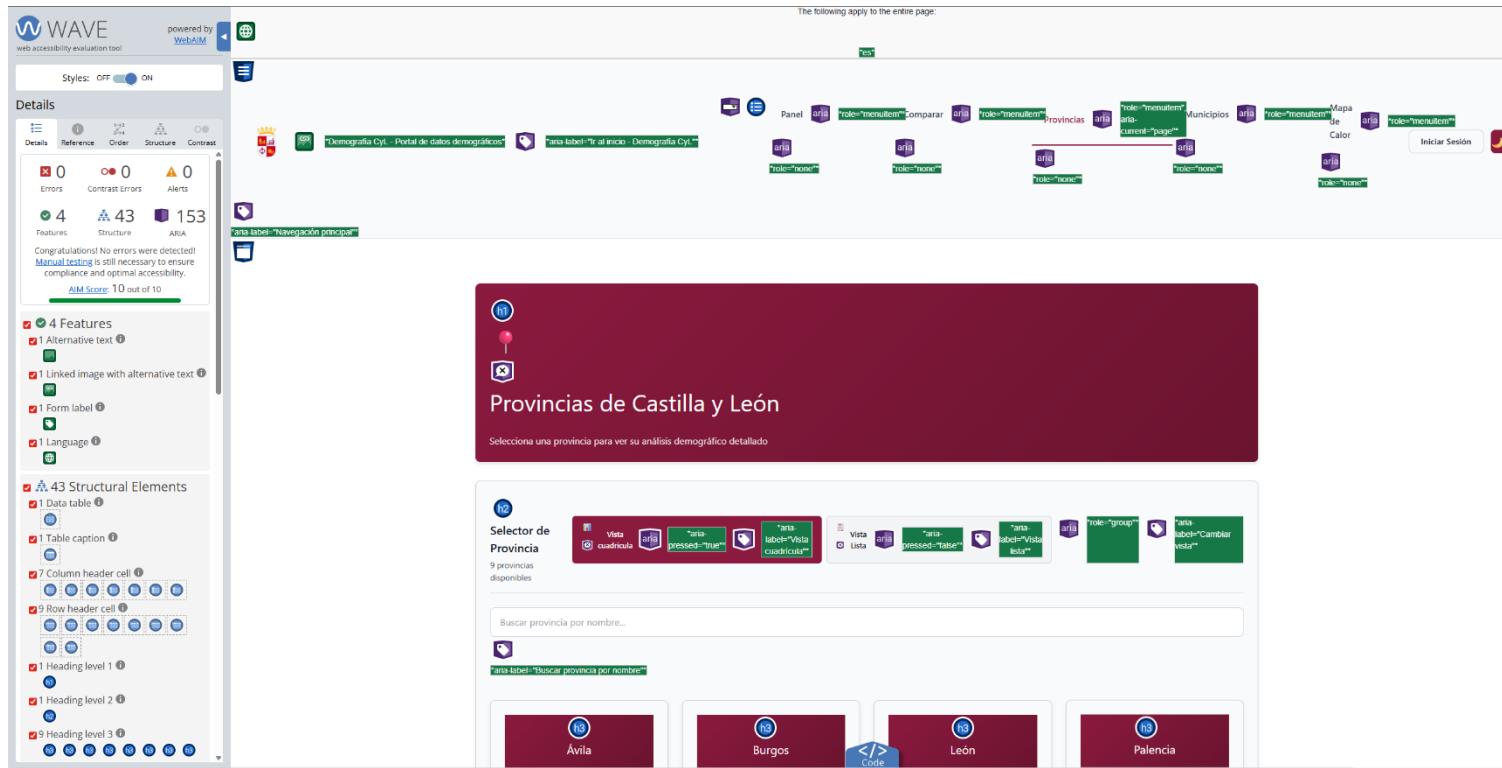


Figura 8. Prueba de usabilidad con wave(página provincias)

## Prueba de Usabilidad

**Perfil de usuario:** Carlos, 32 años, técnico de la Diputación de Valladolid que necesita visualizar geográficamente los datos demográficos.

**Tarea:** "Quiero ver en un mapa dónde hay más defunciones en Castilla y León para identificar zonas críticas."

### Paso 1: Página principal

- "Veo 'Mapa de Calor' en el menú y también hay un botón destacado 'Ver Mapa →'. Bien, es fácil de encontrar."
  - o Acción: Clic en "Mapa de Calor"

### Paso 2: Página del Mapa de Calor

- "Bien, veo controles a la izquierda: puedo elegir año (2020-2023) y tipo de evento (Nacimientos, Defunciones, Matrimonios). También hay un buscador de municipios. El mapa está cargando..."

- "Veo 'Defunciones' como opción. Selecciono eso y año 2022 para ver datos recientes."
- "Mientras carga veo 'Total Regional: Cargando...' - supongo que mostrará el total de defunciones. También hay una leyenda vacía que imagino se llenará con colores."

### Paso 3: Visualización del mapa (Defunciones 2022)

- "¡Ya cargó! Veo el mapa con todos los municipios coloreados. El total regional es 31.017 defunciones en 2022. La leyenda está clara: desde gris (sin datos) hasta rojo oscuro (12+)."
- "Puedo distinguir bien las zonas con más defunciones (rojo intenso) concentradas en el noroeste (León) y algunas zonas del centro. Las zonas grises son municipios sin datos, sobre todo en el este (Soria, Burgos)."
- "El mapa tiene controles de zoom (+/-) y puedo ver el contexto geográfico con las provincias vecinas (Cantabria, La Rioja, Madrid...). Usa Leaflet y OpenStreetMap, buena elección."

## 6. Despliegue

- **Instrucciones de instalación en local:**

Requisitos Previos:

- PHP 8.2 o superior
- Composer
- Node.js 18+ y npm
- MySQL 8.0+ o MariaDB 10.3+
- Git

Instalación Paso a Paso

1. Clonar el Repositorio

```
git clone <https://github.com/samuufr/DatosJuntaCyL>
cd DatosJuntaCyL
```

2. Instalar Dependencias

```
# Dependencias PHP
composer install

# Dependencias Node
npm install
```

### 3. Configurar Entorno

```
# Windows  
copy .env.example .env  
  
# Linux/Mac  
cp .env.example .env
```

Editar .env con tus credenciales de base de datos:

```
DB_DATABASE=datosjuntacyl  
DB_USERNAME=root  
DB_PASSWORD=tu_password
```

### 4. Generar Clave de Aplicación

```
php artisan key:generate
```

6. Crear Tablas y bbdd (si pregunta crear la bbdd decir “yes” y la crea automáticamente)

```
php artisan migrate
```

7. Importar Provincias// Resultado: 9 provincias de Castilla y León

```
php artisan db:seed --class=ProvinciaSeeder
```

8. Importar TODOS los Municipios desde API // Resultado: ~2,249 municipios |  
Tiempo: 1-2 minutos

```
php artisan municipios:import-jcyl
```

9. Importar Usuarios de Prueba

```
php artisan db:seed --class=UsuarioSeeder
```

10. Importar Datos MNP (2020-2023) **⚠ IMPORTANTE:** Ejecutar año por año  
debido a limitaciones de la API (hacerlo todo junto puede dar problemas).

Windows (CMD):

```
php artisan mnp:import --ano-inicio=2020 --ano-fin=2020  
php artisan mnp:import --ano-inicio=2021 --ano-fin=2021  
php artisan mnp:import --ano-inicio=2022 --ano-fin=2022  
php artisan mnp:import --ano-inicio=2023 --ano-fin=2023
```

Windows (PowerShell):

```
2020..2023 | ForEach-Object { php artisan mnp:import --ano-inicio=$_ --ano-fin=$_ }
```

Linux/Mac:

```
for year in 2020 2021 2022 2023; do php artisan mnp:import --ano-inicio=$year --ano-fin=$year; done
```

Resultado esperado:

- ~3,976 registros de nacimientos
- ~6,763 registros de defunciones
- ~2,723 registros de matrimonios
- Total: ~13,462 registros
- Tiempo: 15-20 minutos

#### 11. Importar Población de Municipios (Opcional)

```
php artisan poblacion:importar-api
```

Resultado: Población actualizada de ~2,248 municipios | Tiempo: 1-2 minutos

#### 12. Compilar Assets Frontend ⚠ CRÍTICO (Sin este paso, la aplicación se verá sin estilos CSS.)

```
npm run build
```

#### 13. Iniciar Servidor (aplicación disponible en localhost)

```
php artisan serve
```

#### Verificar Importación

```
# Ver totales
php artisan tinker --execute="echo 'Provincias: ' .
\App\Models\Provincia::count() . PHP_EOL . 'Municipios: ' .
\App\Models\Municipio::count() . PHP_EOL . 'Datos MNP: ' .
\App\Models\DatosMnp::count();"
```

Resultados esperados:

- Provincias: 9
- Municipios: ~2,249
- Datos MNP: ~13,462

- **URLs:**

- Repositorio de GitHub:

<https://github.com/samuuufr/DatosJuntaCyL>

- Pagina desplegada en Railway:

<https://datosjuntacyl-production.up.railway.app>

## 7. Sostenibilidad y "Green Coding"

Justificar las decisiones tomadas para reducir el impacto ambiental:

**Estrategia de Caché:** La decisión de arquitectura más sostenible del proyecto ha sido no consumir la API de la Junta en tiempo real para cada visita.

- **Problema:** Si cada vez que un usuario entra en la web, la aplicación tuviera que consultar la API externa, se generarían miles de peticiones HTTP innecesarias, consumiendo ancho de banda y tiempo de CPU en ambos servidores.
- **Solución "Green":** Se ha implementado un sistema de ingestión y persistencia en base de datos local (MySQL). Los datos se descargan una sola vez mediante los comandos personalizados (php artisan mnp:import) y se sirven desde local.

**Optimización de recursos:** El peso de la página web influye directamente en la energía necesaria para transmitirla y renderizarla.

- **Minificación:** Gracias al uso de **Vite** como empaquetador, todo el código JavaScript y CSS se compila y minifica para producción. Se eliminan espacios en blanco y comentarios, reduciendo el tamaño de los archivos transferidos.
- **Tree-Shaking en CSS:** Al utilizar **Tailwind CSS**, el compilador analiza los archivos HTML/Blade y genera una hoja de estilos que contiene *únicamente* las clases que realmente se están utilizando en el proyecto, descartando todo el CSS sobrante. Esto garantiza que el navegador del usuario no descargue ni procese ni un byte de más.

**Reflexión:** ¿Qué ahorro energético supone el sistema de caché frente a una consulta directa constante?

Esto reduce el tráfico de red drásticamente y convierte consultas complejas de API en consultas SQL indexadas de milisegundos, reduciendo el uso del procesador del servidor

Además...

- Tal y como se observa en el diseño de la interfaz, se ha optado por una paleta de colores oscuros (**Dark Mode** nativo).

- **Justificación:** En dispositivos con pantallas OLED o AMOLED (muy comunes en smartphones actuales), los píxeles negros o muy oscuros están apagados o consumen mucha menos energía que los píxeles blancos brillantes.
- **Beneficio:** Dado que es una aplicación de consulta de datos donde el usuario puede pasar tiempo analizando gráficas, el modo oscuro contribuye directamente a ahorrar batería en el dispositivo del usuario final

- La sostenibilidad del proyecto también contempla la dimensión social y el compromiso ético con la comunidad de desarrollo Open Source.

- **Soporte a la autoría y concienciación:** Para la generación de mapas interactivos se utiliza la librería **Leaflet**. Se ha tomado la decisión consciente de mantener la atribución original que incluye la bandera de Ucrania. Esto responde al llamamiento realizado por su creador, **Volodymyr Agafonkin**, y el equipo de mantenedores, quienes solicitan visibilidad ante la invasión rusa de su país de origen.
- **Alineación con valores humanitarios:** Al preservar este elemento en la interfaz, el proyecto no solo cumple con la licencia de software, sino que actúa como un vector de concienciación pasiva. Se respalda el mensaje "*Appeal to humanity*" (Llamamiento a la humanidad) incluido en la documentación de la herramienta, entendiendo que el desarrollo tecnológico no puede ser ajeno a las realidades sociales y humanitarias de sus creadores.

## 8. Conclusiones

### 8.1. Autoevaluación

- **Dificultades encontradas:**
  - Muchos datos en la api, por lo que con la diferencia de nombres nos costo cargarlos a la bbdd y coordinarlo todo correctamente
  - Problemas con los municipios que se añaden a favoritos
  - En el año 2020 no hay datos de defunciones en salamanca en la api

- Tuvimos que desactivar la verificación SSL en desarrollo debido a que a veces daba errores
- Despliegue con railway y Docker, gran variedad de errores y complicaciones
- **Valoración de la metodología.** Para la coordinación del trabajo hemos usado una metodología que hemos basado en el trabajo conjunto, con la que todos nos sentimos partícipes de todas las partes del proyecto, aportando cada uno nuestro granito de arena y nuestro punto personal. Puede sonar caótico, pero lo hemos realizado de manera ordenada y conjunta

## 8.2. Líneas futuras

- **Propuestas de futuro.**
  - Aunque la API de movimiento natural tiene una cantidad inmensa de datos, nosotros únicamente usamos los relativos a nacimientos, matrimonios y defunciones, a la web se le pueden aplicar muchas mejoras para visualizar todo tipos de datos y filtrados

## 9. Bibliografía

### Referencias Normativas y Herramientas de Validación

- [1] WebAIM, «WAVE web accessibility evaluation tool», 2026. [En línea]. Disponible: <https://wave.webaim.org/>. [Último acceso: enero 2026].

### Fuentes de Datos (Open Data Junta de Castilla y León)

- [2] Junta de Castilla y León, «Movimiento Natural de la Población», Portal de Datos Abiertos. [En línea]. Disponible: [https://datosabiertos.jcyl.es/web/jcyl/set/es/demografia/movimiento\\_poblacion/1284208120307](https://datosabiertos.jcyl.es/web/jcyl/set/es/demografia/movimiento_poblacion/1284208120307).
- [3] Junta de Castilla y León, «Registro de Municipios de Castilla y León», Portal de Datos Abiertos. [En línea]. Disponible: <https://datosabiertos.jcyl.es/web/jcyl/set/es/sector-publico/municipios/1284278782067>.
- [4] Junta de Castilla y León, «API de Análisis de Datos Abiertos - Registro de Municipios», Portal de Datos Abiertos. [En línea]. Disponible: <https://analisis.datosabiertos.jcyl.es/api/explore/v2.1/catalog/datasets/registro-de-municipios-de-castilla-y-leon/records>.

### Documentación Técnica y Tecnologías Utilizadas

- [5] Laravel LLC, «Laravel 11 Documentation», 2026. [En línea]. Disponible: <https://laravel.com/docs/11.x>. (Tecnología base del backend mencionada en la memoria) .
- [6] Tailwind Labs, «Tailwind CSS 4.0 Documentation», 2026. [En línea]. Disponible: <https://tailwindcss.com/docs>. (Framework de estilos mencionado) .
- [7] Chart.js Contributors, «Chart.js User Guide», 2026. [En línea]. Disponible: <https://www.chartjs.org/docs/latest/>. (Librería de gráficos utilizada) .
- [8] V. Agafonkin, «Leaflet - a JavaScript library for interactive maps», 2026. [En línea]. Disponible: <https://leafletjs.com/>. (Librería de mapas y referencia al creador Volodymyr Agafonkin mencionada en sostenibilidad) .

## Artículos y Recursos de Diseño

- [9] Draw.io, «UML Use Case Diagrams with Draw.io», draw.io Blog. [En línea]. Disponible: <https://drawio-app.com/blog/uml-use-case-diagrams-with-draw-io/>