

PROYECTO:

MuseosCyL



Realizado por:

SIMON D.V.

- *Asier De Vega De Vega.*
- *Juan Andrés Simón Graeff.*



Desarrollo de Aplicaciones Web (DAW)

Valladolid, a 5 de febrero de 2026

Contenido

1.	Introducción.....	3
2.	Análisis	3
2.1.	Dataset utilizado	3
3.	Requisitos funcionales	4
3.1.	Diagramas de casos de uso	11
4.	Base de Datos	11
4.1.	Diseño de Base de Datos	11
	Justificación:	12
	Tabla usuarios	12
	Tabla museos	12
	Tabla museos_cache.....	13
	Tabla favoritos.....	14
	Diagrama Entidad-Relación:	15
4.2.	Prototipo de Interfaces	16
4.3.	Guía de Estilos.....	18
5.	Desarrollo.....	20
5.1.	Stack tecnológico	20
	Entorno de desarrollo.....	20
	Backend:	20
	Frontend:	20
	Librerías externas:	20
	API externa:	20
	Arquitectura:.....	20
	Software utilizado.....	20
5.2.	Estructura de carpetas	21
	Descripción de componentes:	21
5.3.	Descripción del funcionamiento	24
	Acceso a la aplicación	24
	Navegación sin registro.....	24

Registro de nuevo usuario	25
Inicio de sesión	26
Exploración con sesión iniciada	26
Vista de detalle del museo	27
Gestión de favoritos.....	28
Cierre de sesión	29
6. Pruebas	29
6.1. Prueba de usabilidad	29
Conclusiones de usabilidad	31
6.2. Análisis de usabilidad.....	31
Páginas analizadas	31
Resultados del análisis.....	31
Análisis de contraste de colores	34
Puntos fuertes de accesibilidad.....	35
Valoración de accesibilidad	35
7. Despliegue	36
7.1. Requisitos previos	36
7.2. Instalación de la aplicación MuseosCyL.....	37
Creación de la base de datos	37
8. Sostenibilidad y "Green Coding"	39
Sostenibilidad y Green Coding.....	39
9. Conclusiones	41
9.1. Autoevaluación	41
9.2. Dificultades Encontradas	41
9.3. Valoración del Trabajo en Equipo	42
9.4. Objetivos Alcanzados.....	42
9.5 Líneas futuras	43
10. Webgrafía.....	44

1. Introducción

El proyecto MuseosCyL se desarrolló como proyecto intermodular, con el objetivo de aplicar de forma práctica los conocimientos adquiridos durante el ciclo formativo de Desarrollo de Aplicaciones Web.

La finalidad principal del proyecto fue la creación de una aplicación web que permitiera consultar información sobre los museos de Castilla y León a través de una API de datos abiertos, así como gestionar museos favoritos por parte de los usuarios registrados.

Para ello, se diseñó y desarrolló una aplicación basada en una arquitectura MVC, combinando el consumo de una API externa con una base de datos propia destinada a la gestión de usuarios y favoritos. De esta forma, se consiguió una aplicación funcional, organizada y acorde a los requisitos establecidos para el proyecto intermodular.

El proyecto se realizó en pareja, fomentando el trabajo colaborativo, la planificación conjunta y el uso de herramientas de control de versiones durante todo el desarrollo.

2. Análisis

2.1. Dataset utilizado

- Nombre: Directorio de Museos de Castilla y León
- URL: [Directorio Museos](#)

Campos utilizados:

- identificador (api_id)
- nombreentidad (nombre)
- localidad
- posicion.lat / posicion.lon (coordenadas)
- descripcionblob
- horario_de_apertura
- servicios_disponibles
- informacion_adicional
- enlace_al_contenido

3. Requisitos funcionales

Listado de las funcionalidades implementadas. Ej. *login*, favoritos, gráficas, etc.

	OBJETIVOS DEL SISTEMA
OBJETIVO	Permitir el registro de nuevos usuarios en el sistema
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Datos Junta Castilla y León
DESCRIPCIÓN	El sistema permitió a los usuarios crear una cuenta introduciendo nombre, correo y contraseña
SUBOBJETIVOS	Facilitar el acceso personalizado a la aplicación
IMPORTANCIA	Alta
URGENCIA	Alta
ESTADO	Implementado
ESTABILIDAD	Alta
COMENTARIOS	Requisito base para el resto de funcionalidades
RF1	ACTORES
ACT-ID	A1
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Departamento de informática
DESCRIPCIÓN	Usuario no registrado que desea crear una cuenta
COMENTARIOS	Necesario para acceder a funcionalidades avanzadas
RF1	REQUISITOS FUNCIONALES
RENDIMIENTO	Registro realizado en tiempo real
IMPORTANCIA	Alta
URGENCIA	Alta
ESTADO	Completado
ESTABILIDAD	Alta
COMENTARIOS	Necesario para acceder a funcionalidades avanzadas

RF2	OBJETIVOS DEL SISTEMA
OBJETIVO	Autenticar a los usuarios registrados
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Datos Junta Castilla y León
DESCRIPCIÓN	El sistema permitió a los usuarios iniciar sesión mediante email y contraseña
SUBOBJETIVOS	Controlar el acceso a la aplicación
IMPORTANCIA	Alta
URGENCIA	Alta
ESTADO	Implementado
ESTABILIDAD	Alta
COMENTARIOS	Necesario para la gestión de favoritos
RF2	ACTORES
ACT-ID	A2
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Departamento de informática
DESCRIPCIÓN	Usuario que accede con sus credenciales
COMENTARIOS	Acceso seguro al sistema
RF2	REQUISITOS FUNCIONALES
RENDIMIENTO	Autenticación inmediata
IMPORTANCIA	Alta
URGENCIA	Alta
ESTADO	Completado
ESTABILIDAD	Alta
COMENTARIOS	Mensajes de error controlados

RF3	OBJETIVOS DEL SISTEMA
OBJETIVO	Mostrar información detallada de los museos
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Datos Junta Castilla y León
DESCRIPCIÓN	El sistema mostró nombre, localidad, ubicación y fotografía de cada museo
SUBOBJETIVOS	Informar al usuario sobre la oferta cultural
IMPORTANCIA	Alta
URGENCIA	Alta
ESTADO	Implementado
ESTABILIDAD	Alta
COMENTARIOS	Información obtenida de fuente externa
RF3	ACTORES
ACT-ID	A3
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Departamento de informática
DESCRIPCIÓN	Usuario registrado o no registrado
COMENTARIOS	Acceso público
RF3	REQUISITOS FUNCIONALES
RENDIMIENTO	Carga optimizada
IMPORTANCIA	Alta
URGENCIA	Alta
ESTADO	Completado
ESTABILIDAD	Alta
COMENTARIOS	Datos actualizados

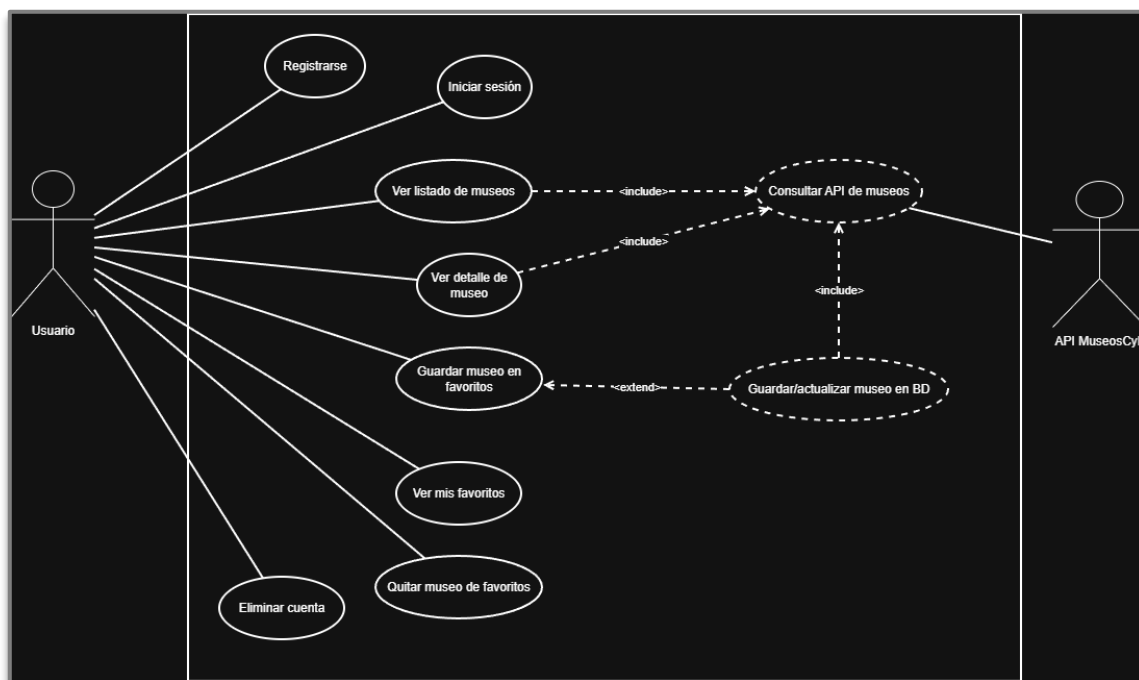
RF4	OBJETIVOS DEL SISTEMA
OBJETIVO	Obtener datos de museos desde una API externa
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Datos Junta Castilla y León
DESCRIPCIÓN	El sistema sincronizó los museos mediante un api id único
SUBOBJETIVOS	Mantener la información actualizada
IMPORTANCIA	Alta
URGENCIA	Media
ESTADO	Implementado
ESTABILIDAD	Media
COMENTARIOS	Dependencia externa
RF4	ACTORES
ACT-ID	A4
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Departamento de informática
DESCRIPCIÓN	API de museos
COMENTARIOS	Fuente oficial
RF4	REQUISITOS FUNCIONALES
RENDIMIENTO	Sincronización eficiente
IMPORTANCIA	Alta
URGENCIA	Media
ESTADO	Completado
ESTABILIDAD	Media
COMENTARIOS	Control de duplicados mediante api_id

RF5	OBJETIVOS DEL SISTEMA
OBJETIVO	Permitir guardar museos favoritos
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Datos Junta Castilla y León
DESCRIPCIÓN	El sistema permitió a los usuarios guardar museos en su lista personal
SUBOBJETIVOS	Facilitar el acceso rápido a museos de interés
IMPORTANCIA	Alta
URGENCIA	Media
ESTADO	Implementado
ESTABILIDAD	Alta
COMENTARIOS	Funcionalidad dependiente del inicio de sesión
RF5	ACTORES
ACT-ID	A2
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Departamento de informática
DESCRIPCIÓN	Usuario autenticado que gestiona sus museos favoritos
COMENTARIOS	Debe haber iniciado sesión previamente
RF5	REQUISITOS FUNCIONALES
RENDIMIENTO	La acción de añadir a favoritos se realizó de forma inmediata
IMPORTANCIA	Alta
URGENCIA	Media
ESTADO	Completado
ESTABILIDAD	Alta
COMENTARIOS	Se verificó que el museo existiera en el sistema

RF6	OBJETIVOS DEL SISTEMA
OBJETIVO	Permitir eliminar museos de la lista de favoritos
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Datos Junta Castilla y León
DESCRIPCIÓN	El sistema permitió a los usuarios eliminar museos previamente guardados en su lista de favoritos
SUBOBJETIVOS	Mantener la lista de favoritos actualizada
IMPORTANCIA	Media
URGENCIA	Media
ESTADO	Implementado
ESTABILIDAD	Alta
COMENTARIOS	Acción reversible volviendo a añadir el museo
RF6	ACTORES
ACT-ID	A2
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Departamento de informática
DESCRIPCIÓN	Usuario autenticado que administra su lista de favoritos
COMENTARIOS	Interacción directa desde la lista de favoritos
RF6	REQUISITOS FUNCIONALES
RENDIMIENTO	Eliminación realizada de manera inmediata
IMPORTANCIA	Media
URGENCIA	Media
ESTADO	Completado
ESTABILIDAD	Alta
COMENTARIOS	No afectó a otros usuarios

RF7	OBJETIVOS DEL SISTEMA
OBJETIVO	Mostrar los museos marcados como favoritos por el usuario
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Datos Junta Castilla y León
DESCRIPCIÓN	El sistema mostró exclusivamente los museos que cada usuario había marcado como favoritos
SUBOBJETIVOS	Permitir una visualización clara y personalizada
IMPORTANCIA	Alta
URGENCIA	Media
ESTADO	Implementado
ESTABILIDAD	Alta
COMENTARIOS	Vista personalizada por usuario
RF7	ACTORES
ACT-ID	A2
VERSIÓN	1.0
AUTORES	Juan Andrés Simón Graeff, Asier De Vega De Vega
FUENTES	Departamento de informática
DESCRIPCIÓN	Usuario autenticado que consulta sus museos favoritos
COMENTARIOS	Acceso restringido a sus propios datos
RF7	REQUISITOS FUNCIONALES
RENDIMIENTO	Listado cargado de forma rápida
IMPORTANCIA	Alta
URGENCIA	Media
ESTADO	Completado
ESTABILIDAD	Alta
COMENTARIOS	No se mostraron museos no marcados

3.1. Diagramas de casos de uso



8

En el diagrama se muestran las principales funcionalidades de la aplicación y cómo interactúan los actores. Tanto el listado de museos como el detalle dependen de la consulta a la API, lo que se representa mediante relaciones *include*. Cuando el usuario guarda un museo en favoritos, la aplicación extiende esa acción para almacenar o actualizar la información en la base de datos. También se reflejan las acciones relacionadas con la gestión del usuario, como ver y eliminar favoritos o borrar la cuenta.

4. Base de Datos

4.1. Diseño de Base de Datos

La base de datos del proyecto se diseñó con el objetivo de gestionar los datos de los usuarios, los museos marcados como favoritos y mantener una caché local del catálogo completo de museos. Es un sistema híbrido que combina base de datos local para consultas de lectura con sincronización periódica mediante script, reduciendo la dependencia de la API externa y mejorando significativamente el rendimiento.

Durante el desarrollo del proyecto se trabajó con dos fuentes de datos diferenciadas. Por un lado, la API de Museos de Castilla y León, utilizada como fuente de datos para la sincronización periódica del catálogo completo de 172 museos. Por otro lado, la base de datos local MySQL, utilizada para consultar el listado de museos, almacenar usuarios registrados y gestionar favoritos.

Este planteamiento arquitectónico reduce las llamadas a servicios externos durante la navegación, mejora el funcionamiento ante caídas de la API externa, permite controlar la fecha de actualización del dataset e incrementa la velocidad de respuesta. La base de datos es altamente escalable y puede soportar miles de usuarios sin degradación de rendimiento.

Justificación:

La base de datos se denomina museoscyl y está compuesta por cuatro tablas principales:

- usuarios
- museos
- museos_caché
- favoritos

Cada una de estas tablas cumple una función concreta dentro del funcionamiento de la aplicación.

Tabla usuarios

La tabla usuarios almacena la información de las personas registradas en la aplicación. Los campos principales son:

Campo	Tipo	Restricciones	Descripción
id	INT	PK, AUTO_INCREMENT	Identificador único del usuario
nombre	VARCHAR(100)	NOT NULL	Nombre completo del usuario
email	VARCHAR(100)	NOT NULL, UNIQUE	Correo electrónico (login)
password	VARCHAR(255)	NOT NULL	Contraseña hasheada (password_hash)

9

Esta tabla permite identificar a cada usuario y asociarlo con sus museos favoritos.

Tabla museos

La tabla museos se diseñó para almacenar únicamente aquellos museos que han sido marcados como favoritos por algún usuario.

La información de los museos se obtiene del almacenamiento local (museos_caché), que a su vez se sincroniza desde la API de datos abiertos de Castilla y León.

Los campos principales son:

Campo	Tipo	Restricciones	Descripción
id	INT	PK, AUTO_INCREMENT	Identificador único interno
api_id	VARCHAR(50)	NOT NULL, UNIQUE	ID del museo en la API
nombre	VARCHAR(255)	NOT NULL	Nombre del museo
localidad	VARCHAR(100)	-	Municipio donde se ubica
latitud	DECIMAL(10,8)	-	Coordenada GPS (latitud)
longitud	DECIMAL(11,8)	-	Coordenada GPS (longitud)
horario	TEXT	-	Horario apertura
url	VARCHAR(255)	-	Enlace a la web oficial
información_adicional	TEXT	-	Datos adicionales

10

El campo api_id permitió evitar duplicidades y relacionar correctamente los datos de la API con los datos almacenados localmente.

Tabla museos_cache

La tabla museos_cache almacena el catálogo completo de museos obtenido desde la API. Se actualiza mediante un proceso de sincronización que se ejecuta de forma periódica.

Los datos provienen de la API REST de Datos Abiertos de la Junta de Castilla y León:

[Datos API](#)

Campos utilizados de la API:

- identificador → api_id
- nombreentidad → nombre
- localidad → localidad
- horario_de_apertura → horario (limpiando HTML con strip_tags)
- enlace_al_contenido → url
- posicion.lat → latitud
- posicion.lon → longitud
- informacion_adicional → informacion_adicional (para extraer tipología)

Proceso de sincronización:

El script sync_museos.php ejecuta las siguientes operaciones:

1. Descarga todos los museos de la API usando paginación (100 registros por petición)
2. Extrae y limpia los campos HTML (horario viene con tags , , <p>)
3. Busca URLs en el campo enlace_al_contenido
4. Extrae información adicional del museo mediante expresiones regulares
5. Limpia la tabla museos_cache con TRUNCATE
6. Inserta los 172 museos procesados

Campo	Tipo	Restricciones	Descripción
id	INT	PK, AUTO_INCREMENT	Identificador único interno
api_id	VARCHAR(50)	NOT NULL, UNIQUE	ID del museo en la API
nombre	VARCHAR(255)	NOT NULL	Nombre del museo
localidad	VARCHAR(100)	-	Municipio donde se ubica
tipologia	VARCHAR(100)	-	Tipo de museo (Arqueología, Arte, etc.)
latitud	DECIMAL(10,8)	-	Coordenada GPS (latitud)
longitud	DECIMAL(11,8)	-	Coordenada GPS (longitud)
horario	TEXT		Horario de apertura extraído del JSON
url	VARCHAR(255)		Enlace a la web oficial del museo
Información_adicional	TEXT		Datos adicionales en formato text
ultima_actualizacion	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Fecha de última sincronización

Tabla favoritos

La tabla favoritos actuó como tabla intermedia entre usuarios y museos, representando una relación de muchos a muchos.

Un usuario puede marcar varios museos como favoritos y un mismo museo puede ser marcado por varios usuarios.

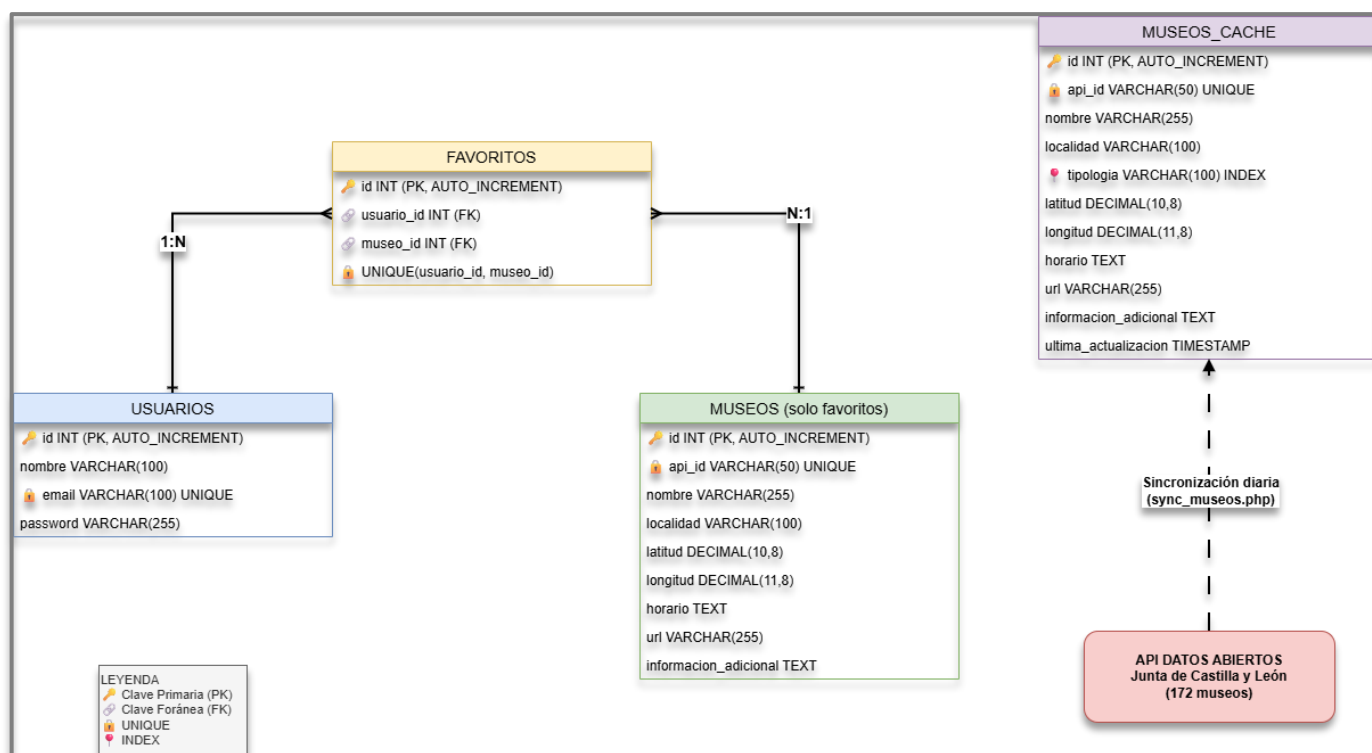
Los campos principales son:

Campo	Tipo	Restricciones	Descripción
id	INT	PK, AUTO_INCREMENT	Identificador único
usuario_id	INT	NOT NULL, FK → usuarios(id)	ID del usuario
museo_id	INT	NOT NULL, FK → museos(id)	ID del museo

12

Se utilizó una clave única en usuario_id y museo_id para evitar que un mismo usuario pudiera marcar el mismo museo más de una vez. Comportamiento ON DELETE CASCADE, Si se elimina un usuario, todos sus favoritos se eliminan automáticamente. Si se elimina un museo, también se eliminan las relaciones asociadas.

Diagrama Entidad-Relación:

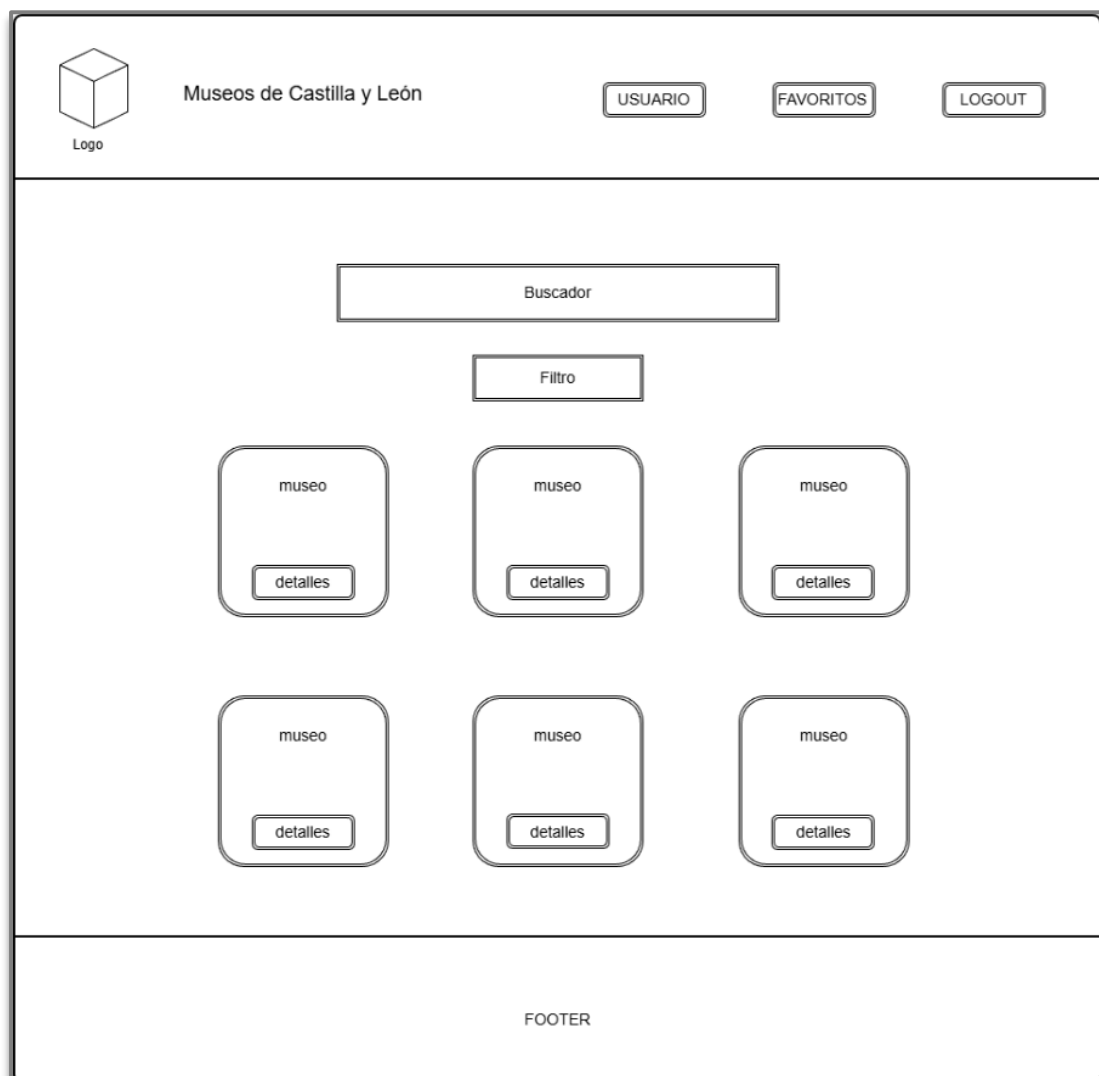


13

4.2. Prototipo de Interfaces

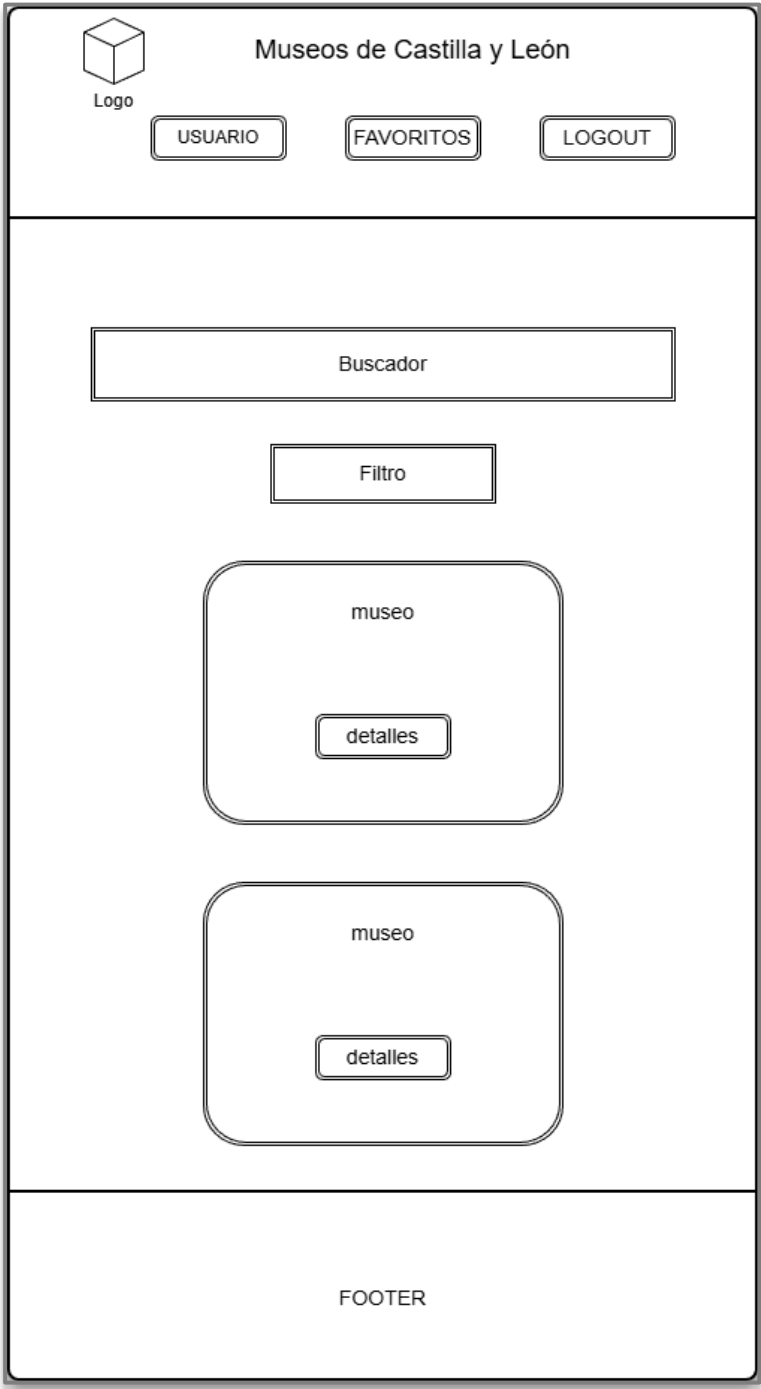
Prototipo versión escritorio

Se han diseñado prototipos de fidelidad baja en la aplicación Draw.io, centrándose en la distribución de contenidos (*layout*) y la jerarquía visual. Se han creado dos maquetas, aplicando el concepto de Diseño Adaptativo (*Responsive Design*) para las versiones de Escritorio y Smartphone.



La página de inicio en versión Escritorio presenta un layout con áreas de contenido definidas. En la cabecera está el Logo posicionado a la izquierda con el título de la web. Cuerpo central con el formulario de login centrado con un patrón que dirige la atención del usuario inmediatamente al elemento más importante de la página. Pie de página con enlace a Inicio y Mis Favoritos.

Prototipo versión móvil



La versión móvil de la página de inicio adapta el contenido a pantallas reducidas, manteniendo la funcionalidad completa, pero optimizando el espacio vertical para que el usuario no tenga que hacer scroll horizontal. La versión smartphone presenta cabecera y footer reducidos por la necesidad de maximizar el espacio disponible para el contenido principal, que en pantallas pequeñas es un recurso limitado.

4.3. Guía de Estilos

Esta guía de estilos documenta las decisiones visuales tomadas durante el desarrollo, estableciendo los criterios que garantizan la consistencia del diseño en toda la aplicación.

Logotipo

El proceso de diseño comenzó con la creación del logotipo mediante herramientas de generación de imágenes online, el objetivo era reflejar el patrimonio histórico y cultural de Castilla y León.



16

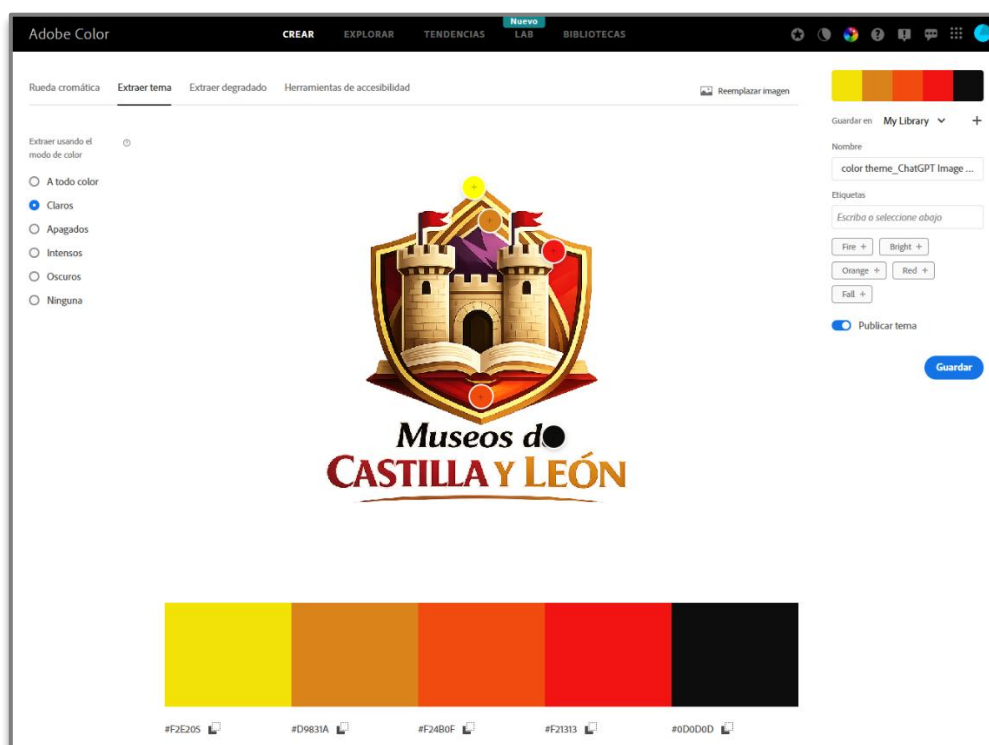
Paleta de colores

La paleta cromática del sistema se extrajo directamente del logotipo utilizando Adobe Color, herramienta profesional de generación de paletas cromáticas, asegurando coherencia cromática y armonía visual en todos los elementos de la web.

El color primario es el rojo (#F23A1F), utilizado en botones principales, títulos, enlaces y bordes destacados. Para los textos se utiliza negro (#0D0D0D) garantizando legibilidad y contraste, mientras que el blanco (#FFFFFF) se aplica en fondos principales y contenedores aportando limpieza visual.

El header de la aplicación utiliza un degradado horizontal que combina estos colores, comenzando en amarillo, pasando por naranja y finalizando en rojo, creando una transición visual coherente con la paleta establecida.

18



17

Diseño responsive

La aplicación se adapta a diferentes dispositivos mediante un diseño responsive. El grid de museos utiliza auto-fill con un mínimo de 320px por tarjeta, ajustándose automáticamente al ancho de pantalla disponible, lo que permite mostrar entre 3-4 columnas en escritorio, 2 en tablet y 1 en dispositivos móviles.

Principios aplicados

El diseño se fundamenta en la búsqueda constante de la simplicidad y la funcionalidad. Se ha mantenido la consistencia visual aplicando los mismos colores, tipografías y espaciados en toda la aplicación para que el usuario reconozca fácilmente los elementos. El contraste garantiza que todos los textos y elementos sean legibles. El diseño es limpio y directo, sin elementos decorativos innecesarios que puedan distraer del objetivo principal: facilitar el acceso a la información sobre los museos. La jerarquía visual se establece mediante el uso estratégico de tamaños y colores, guiando naturalmente la atención del usuario hacia los elementos más importantes de cada página. Todas estas decisiones se han tomado pensando en que la aplicación sea fácil de usar para cualquier persona, independientemente del dispositivo que utilice o de sus capacidades.

5. Desarrollo

5.1. Stack tecnológico

Entorno de desarrollo

El proyecto ha sido desarrollado utilizando las siguientes tecnologías:

Backend:

- PHP: Lenguaje de programación del lado del servidor para la lógica de negocio y gestión de datos.
- MySQL: Sistema de gestión de bases de datos relacional para el almacenamiento persistente de información.
- PDO (PHP Data Objects): Capa de abstracción para el acceso a la base de datos, utilizada para realizar consultas preparadas y prevenir inyecciones SQL.

Frontend:

- HTML5: Marcado semántico de la estructura de las páginas web.
- CSS3: Hojas de estilo para el diseño visual y responsive de la aplicación.
- JavaScript: Lenguaje de programación del lado del cliente para interactividad, filtrado en tiempo real y peticiones AJAX.

Librerías externas:

- Leaflet: Librería JavaScript para mapas interactivos, utilizada para mostrar la ubicación geográfica de los museos.

API externa:

- API de Datos Abiertos de Castilla y León: Servicio REST que proporciona información actualizada sobre los museos de la comunidad autónoma.
 - [Datos Abiertos](#)

Arquitectura:

- Patrón MVC (Modelo-Vista-Controlador) personalizado: Implementación propia sin frameworks externos, garantizando una arquitectura limpia, modular y escalable.

Software utilizado

- XAMPP Portable: Servidor local Apache + MySQL para el desarrollo y pruebas.
- Visual Studio Code: Editor de código fuente principal.
- phpMyAdmin: Interfaz web para la administración de la base de datos MySQL.
- Git 2.43: Sistema de control de versiones para el seguimiento de cambios en el código.
- GitHub: Plataforma de alojamiento del repositorio remoto.

- Google Chrome: Herramientas de Desarrollador del navegador para depuración y debugging de JavaScript y CSS.

5.2. Estructura de carpetas

museoscyl/	
├ index.php	→ Punto de entrada y router principal
├ sync_museos.php	→ Script para sincronizar/actualizar museos desde la API a la BD
├ museoscyl.sql	→ Script SQL para crear la base de datos y tablas
├ README.md	→ Documentación del proyecto
├ .gitignore	→ Archivo para que no se suba a git archivo con claves
├ cron.bat	→ Archivo ejecutable por el programador de tareas para que se actualice la web 1 vez al día
├ config/	→ Configuración del proyecto
│ └ config.php	→ Constantes globales (BASE_URL, API, nombre del proyecto, etc.)
│ └ basedatos.php	→ Conexión a MySQL con PDO
│ └ recaptcha.php	→ Archivo de guardado de claves por .gitignore
├ controllers/	→ Controladores (lógica)
│ └ ControladorMuseo.php	→ Listado, filtros y detalle de museos
│ └ ControladorUsuario.php	→ Registro, login y logout
│ └ ControladorFavorito.php	→ Guardar/eliminar/listar favoritos
├ models/	→ Modelos (acceso a datos)
│ └ Museo.php	→ Gestión de museos (API/BD según versión)
│ └ Usuario.php	→ Gestión de usuarios (registro/login)
│ └ Favorito.php	→ Operaciones con la tabla de favoritos
├ views/	→ Vistas (pantallas)
│ └ layout/	
│ │ └ header.php	→ Cabecera común
│ │ └ nav.php	→ Menú superior con logo
│ │ └ footer.php	→ Pie de página
│ └ museos/	
│ │ └ listado.php	→ Listado de museos + buscador/filtros
│ │ └ detalle.php	→ Detalle del museo (horario, web y mapa)
│ └ usuarios/	
│ │ └ login.php	→ Formulario de inicio de sesión
│ │ └ registro.php	→ Formulario de registro
│ │ └ favoritos.php	→ Listado de museos favoritos
├ public/	→ Archivos públicos
├ img/	
│ └ logo.png	→ Logo del proyecto
├ js/	
└ favoritos.js	→ JS para gestionar favoritos (AJAX)

18

El proyecto MuseosCyL sigue el patrón arquitectónico **Modelo-Vista-Controlador (MVC)** implementado desde cero sin frameworks externos. Esta decisión de diseño permite un control total sobre el código y facilita la comprensión de los fundamentos del patrón MVC.

Descripción de componentes:

index.php - Punto de entrada único y router principal. Recibe peticiones mediante el parámetro GET 'accion' y las redirige al controlador correspondiente, gestionando el flujo de navegación de toda la aplicación.

sync_museos.php - Script de sincronización que descarga los 172 museos desde la API mediante peticiones paginadas, procesa los datos HTML con expresiones regulares y actualiza la tabla museos_cache..

museoscyl.sql - Script SQL con la estructura completa de la base de datos: tablas, campos, claves primarias y foráneas, índices y restricciones de integridad. Permite recrear la base de datos completa desde cero.

README.md - Documentación técnica con instrucciones de instalación, requisitos del sistema, configuración necesaria y guía de uso básica.

.gitignore - Excluye archivos sensibles del control de versiones, especialmente recaptcha.php que contiene claves secretas, además de archivos temporales y configuraciones del IDE.

cron.bat - Script ejecutable para el Programador de tareas de Windows que automatiza la sincronización diaria. Registra la salida en log para monitorizar el proceso.

Directorio config/

config.php - Define constantes globales: BASE_URL, API_MUSEOS_URL, configuración de base de datos y nombre del proyecto. Centraliza la configuración para facilitar el despliegue en diferentes entornos.

basedatos.php - Clase de conexión a MySQL mediante PDO. Implementa Singleton para una única conexión activa, configurada con prepared statements, manejo de excepciones y charset UTF-8.

recaptcha.php - Almacena las claves públicas y privadas de Google reCAPTCHA v2. Excluido de Git por .gitignore. Incluye la función de verificación del token en el servidor.

Directorio controllers/

ControladorMuseo.php - Gestiona la visualización de museos. El método listar() obtiene el catálogo desde BD local, recupera favoritos del usuario y carga la vista con grid y filtros. El método detalle() busca un museo específico y muestra información completa con mapa.

ControladorUsuario.php - Implementa autenticación de usuarios. Procesa registro validando email único, verificando reCAPTCHA y hasheando contraseñas. Gestiona login verificando credenciales con password_verify() e inicia sesiones PHP. El método logout() destruye la sesión activa.

ControladorFavorito.php - Gestiona favoritos. El método agregar() guarda el museo si no existe y crea la relación evitando duplicados. El método eliminar() borra la relación y listar() muestra todos los favoritos del usuario con sus datos completos.

Directorio models/

Museo.php - Encapsula acceso a datos de museos. obtenerTodos() consulta museos_cache adaptando el formato BD al esperado por las vistas. obtenerPorApild() busca mediante índice optimizado. guardarMuseoSiNoExiste() inserta en tabla museos incluyendo horario, URL e información adicional.

Usuario.php - Gestiona usuarios con consultas preparadas PDO. buscarPorEmail() retorna el usuario si existe. autenticar() verifica contraseña hasheada. crear() valida email único, hashea contraseña con BCrypt e inserta el registro.

Favorito.php - Opera sobre la tabla favoritos. agregar() inserta relación usuario-museo protegido por UNIQUE. eliminar() borra relación validando propiedad. listarPorUsuario() hace JOIN con museos para información completa. obtenerApildsFavoritos() devuelve array de IDs para marcar visualmente en el listado.

*Directorio views/**layout/*

header.php - Cabecera HTML común.

nav.php - Barra de navegación con logo y menú adaptativo. Muestra opciones según estado de sesión: usuario logueado ve Inicio, Favoritos y Cerrar Sesión; sin sesión ve Inicio, Registrarse e Iniciar Sesión.

footer.php - Pie de página

Subdirectorio museos/

listado.php - Catálogo completo en grid responsive de tarjetas. Incluye buscador en tiempo real que filtra por nombre/localidad, selector de tipología y contador dinámico. Cada tarjeta muestra nombre, localidad, tipo y botón de favorito.

detalle.php - Información completa del museo. Extrae horario con expresiones regulares, muestra enlace web oficial y mapa interactivo.

Subdirectorio usuarios/

login.php - Formulario de inicio de sesión con email, contraseña y reCAPTCHA v2. Validación en frontend y completa en backend. Muestra mensajes de error.

registro.php - Formulario de creación de cuenta con nombre, email, contraseña y confirmación. Valida coincidencia de contraseñas, verifica reCAPTCHA y comprueba email único antes de hashear y almacenar.

favoritos.php - Lista museos favoritos en tarjetas simplificadas con nombre, localidad y tipología. Incluye botones "Ver detalle" y "Eliminar" con confirmación. Muestra mensaje si no hay favoritos.

Directorio public/

img/logo.png - Logotipo del proyecto optimizado para web en formato PNG con transparencia, utilizado en la barra de navegación.

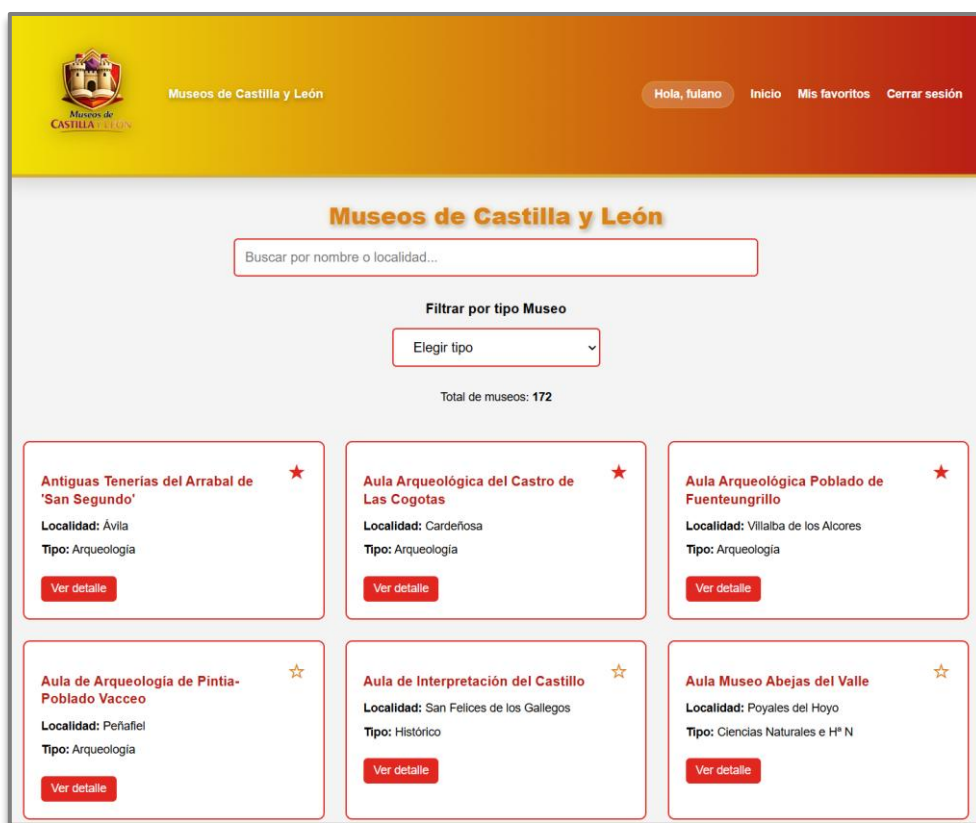
js/favoritos.js - Implementa añadir/eliminar favoritos sin recargar página. La función toggleFavorito() envía petición AJAX POST con datos JSON, actualiza icono de estrella según respuesta y gestiona errores con alertas.

5.3. Descripción del funcionamiento

Acceso a la aplicación

Para acceder a la aplicación, el usuario debe abrir un navegador web y navegar a la URL <http://localhost/museoscyl/>

La página de inicio muestra un listado con los museos de Castilla y León obtenidos desde la API de Datos Abiertos.



19

Navegación sin registro

El usuario puede explorar los museos sin necesidad de crear una cuenta:

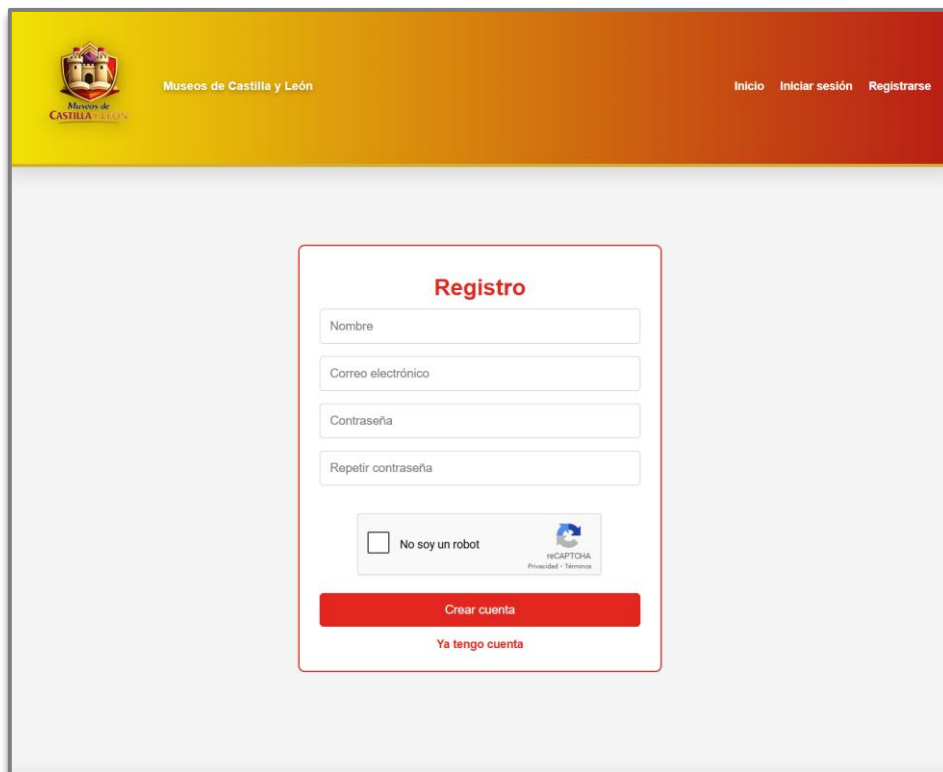
- Buscador: Permite filtrar museos escribiendo en el campo de texto. La búsqueda se realiza sobre el nombre del museo y la localidad.
- Filtro por tipo: Desplegable que permite seleccionar una tipología específica (Arqueología, Arte, Etnología, etc.). Puede combinarse con el buscador para refinar resultados.
- Información visible: Cada tarjeta muestra:
 - Nombre del museo
 - Tipo de museo
 - Localidad
 - Coordenadas geográficas
 - Añadir a favoritos ☆

- Botón "Ver detalle": Al hacer clic sin estar logueado, redirige automáticamente a la página de inicio de sesión.

Registro de nuevo usuario

Para acceder a funcionalidades avanzadas, el usuario debe registrarse:

- Hacer clic en "Registrarse" en el menú superior.
- Completar el formulario con:
 - Nombre completo
 - Correo electrónico (único en el sistema)
 - Contraseña (mínimo 6 caracteres)
 - Confirmación de contraseña
- Hacer clic en el botón "Registrarse".
- Si el registro es exitoso, el sistema muestra un mensaje de confirmación y redirige al login.



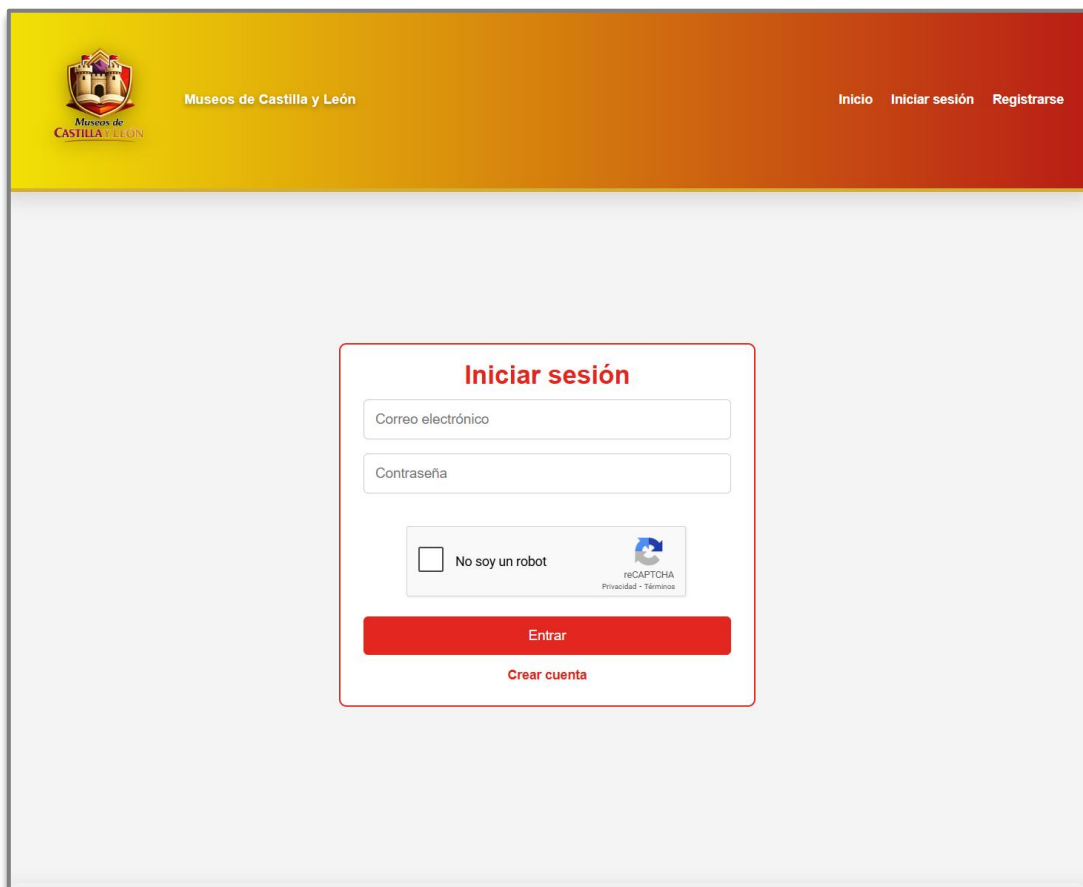
The screenshot shows the registration page of the Museums of Castilla y León website. The header is a gradient of yellow and orange, featuring the museum's logo on the left and navigation links "Inicio", "Iniciar sesión", and "Registrarse" on the right. The main content area is light gray and contains a white registration form titled "Registro" in red. The form includes four input fields: "Nombre", "Correo electrónico", "Contraseña", and "Repetir contraseña". Below these fields is a checkbox labeled "No soy un robot" next to a reCAPTCHA logo. At the bottom of the form, there is a red button labeled "Crear cuenta" and a red link labeled "Ya tengo cuenta".

20

En el Formulario de Registro, todos los campos son obligatorios, el correo debe tener formato válido, las contraseñas deben coincidir, la contraseña se almacena hasheada.

Inicio de sesión

- Hacer clic en "Iniciar sesión" en el menú superior.
- Introducir:
 - Correo electrónico
 - Contraseña
- Hacer clic en "Entrar".
- Si las credenciales son correctas, el sistema crea una sesión y redirige al listado de museos.
- El menú superior muestra: "Hola, [Nombre]" y las opciones "Mis Favoritos" y "Cerrar sesión".



Museos de Castilla y León

Inicio Iniciar sesión Registrarse

Iniciar sesión

Correo electrónico

Contraseña

☐ No soy un robot

reCAPTCHA
Privacidad - Términos

Entrar

Crear cuenta

21

Exploración con sesión iniciada

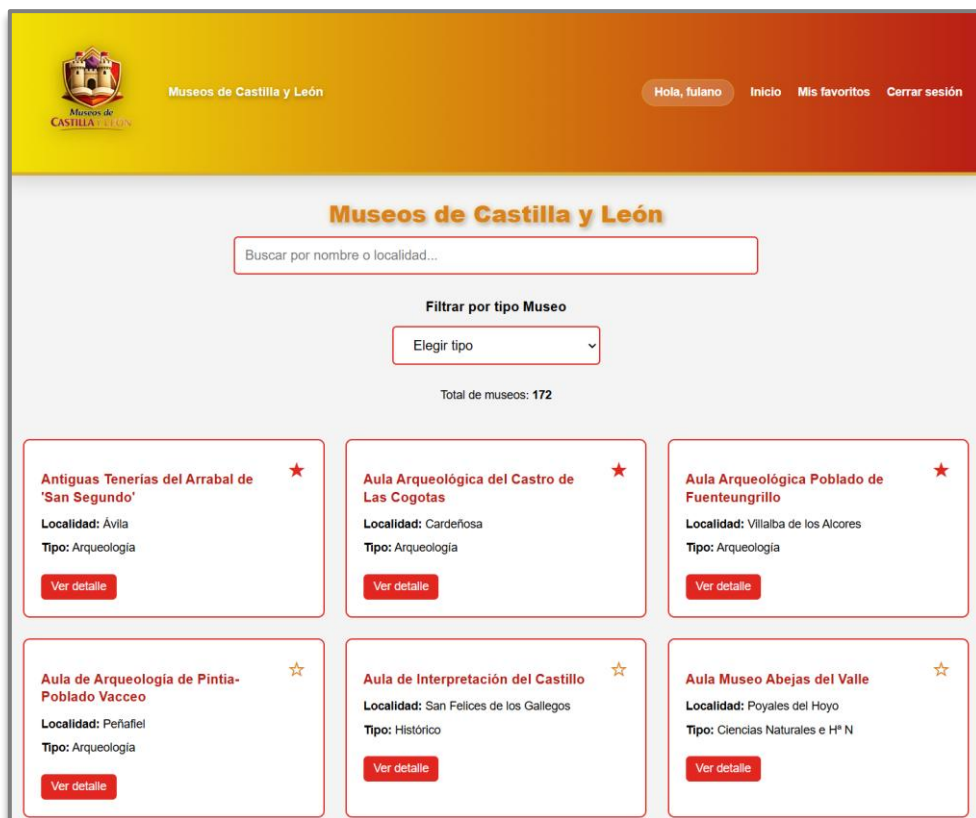
Una vez autenticado, el usuario tiene acceso completo:

Listado de museos:

- Cada tarjeta muestra una estrella (☆):

26

- Blanca (☆): Museo no guardado en favoritos.
- Roja (★): Museo ya guardado en favoritos.
- Clic en la estrella: Añade o elimina el museo de favoritos, sin recargar la página.
- Botón "Ver detalle": Lleva a la ficha completa del museo.



22

Vista de detalle del museo

Al hacer clic en "Ver detalle", se muestra:

- Nombre del museo: Título principal en rojo.
- Localidad: Ciudad donde se encuentra.
- Horario de apertura: Días y horas de visita.
- Enlace a página oficial: Botón rojo que abre la web del museo en una nueva pestaña.
- Mapa interactivo: Visualización con Leaflet mostrando la ubicación exacta con un marcador. El mapa permite zoom y desplazamiento.
- Botón "Volver al listado": Regresa a la página principal.



Museos de Castilla y León

Hola, fulano Inicio Mis favoritos Cerrar sesión

MUSAC. Museo de Arte Contemporáneo de Castilla y León

Localidad: León

Horario: Del 01/01 al 31/12 - De Martes a Domingo - De 11:00 a 14:00 y de 17:00 a 20:00 y 31 de diciembre abierto en horario de mañanaInformación sobre los días de cierre01/01, 25/12Lunes

Web oficial: [Abrir página del museo](#)

Ubicación

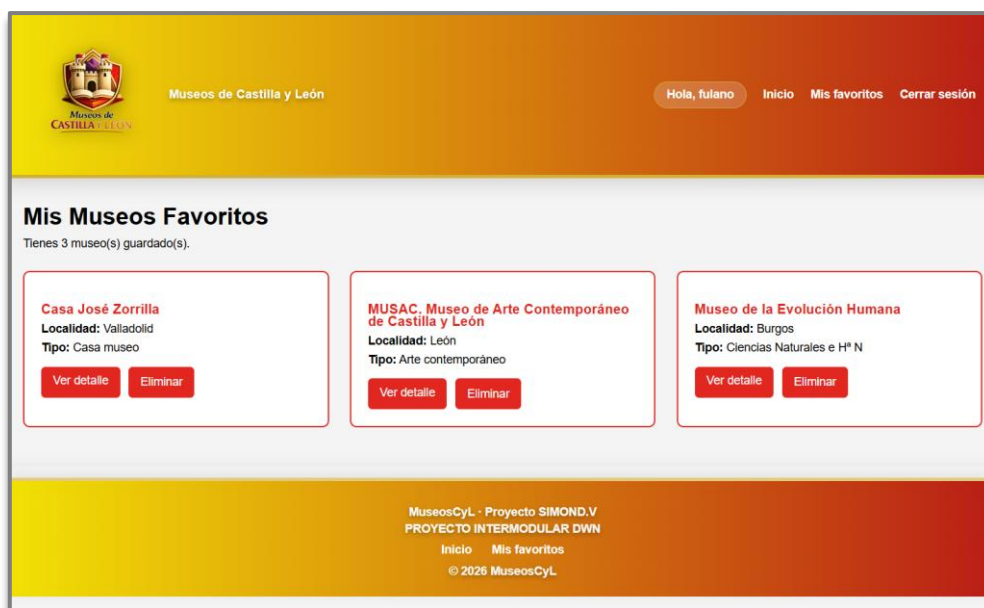


[← Volver al listado](#)

23

Gestión de favoritos

- Hacer clic en "Mis Favoritos" en el menú superior.
- Se muestra un grid con todos los museos favoritos del usuario.
- Cada tarjeta incluye:
 - Nombre, localidad y coordenadas.
 - Botón "Ver detalles": Accede a la ficha completa.
 - Botón "Eliminar": Pide confirmación y elimina el museo de favoritos.



24

Si no hay favoritos guardados, se muestra el mensaje: *"No tienes museos guardados."*

Cierre de sesión

- Hacer clic en "Cerrar sesión" en el menú superior.
- El sistema destruye la sesión y redirige al listado de museos.
- El menú vuelve a mostrar las opciones de usuario no autenticado.

6. Pruebas

6.1. Prueba de usabilidad

Se ha realizado una prueba de usabilidad con un usuario real para evaluar la facilidad de uso y la intuitividad de la aplicación MuseosCyL.

Perfil del usuario

- Edad: 38 años
- Nivel tecnológico: Medio (usa ordenador diariamente pero no es desarrollador)
- Experiencia previa: Primera vez usando la aplicación
- Contexto: Usuario interesado en visitar museos de la localidad de León

Tareas asignadas

Se solicitó al usuario realizar las siguientes tareas sin asistencia:

Tarea 1: Registro en la aplicación

Tiene como objetivo crear una cuenta de usuario nueva. Para ello el usuario necesita:

- Localizar el botón "Registrarse"
- Completar el formulario de registro
- Confirmar el registro exitoso

Tarea 2: Búsqueda de museos de arqueología en León

Tiene como objetivo encontrar museos de arqueología ubicados en la localidad de León. Pasos esperados:

- Utilizar el buscador escribiendo "León"
- Seleccionar el filtro "Arte Contemporáneo"
- Revisar los resultados filtrados

Tarea 3: Guardar un museo en favoritos y acceder a su detalle

Tiene como objetivo marcar un museo como favorito y ver su información completa

- Hacer clic en la estrella de un museo
- Acceder a "Mis Favoritos" desde el menú
- Hacer clic en "Ver detalles"
- Visualizar el mapa de ubicación

Resultados

Tarea	Tiempo	Éxito	Observaciones
Tarea 1	3 min	Sí	Completada sin problemas. El usuario encontró el enlace fácilmente.
Tarea 2	1 min	Sí	Filtrado intuitivo. El usuario combinó ambos filtros sin dificultad.
Tarea 3	1:10 min	Sí	El usuario comprendió el sistema de estrellas inmediatamente.

25

Tiempo total: 6 minutos.

Feedback del usuario

Aspectos positivos:

- "La búsqueda es muy rápida, se filtra al momento"
- "Los colores rojos destacan bien los botones importantes"
- "El mapa es útil para saber exactamente dónde está el museo"
- "Las estrellas ayudan a marcar favoritos"

Aspectos a mejorar:

- "Estaría bien poder ver los museos en un mapa general, no solo de uno en uno"
- "Me gustaría poder filtrar por provincia además de por tipo"

Conclusiones de usabilidad

- Navegación intuitiva: El usuario completó todas las tareas sin ayuda.
- Claridad visual: La paleta de colores (#E12720, #FFFFFF, #000000) proporciona buen contraste.
- Feedback inmediato: Las acciones AJAX funcionan sin recargas, mejorando la experiencia.
- Mejora sugerida: Implementar un mapa global con todos los museos (funcionalidad futura).

Valoración global: 8.5/10

6.2. Análisis de usabilidad

Se ha realizado un análisis de accesibilidad utilizando la herramienta WAVE Evaluation Tool, con el objetivo de evaluar el cumplimiento de los estándares WCAG 2.1 (Web Content Accessibility Guidelines).

Páginas analizadas

Se analizaron las siguientes páginas de la aplicación:

- Página de inicio (listado de museos) - <http://localhost/museoscyl/index.php>
- Página de registro - <http://localhost/museoscyl/index.php?accion=registro>
- Página de detalle de museo
<http://localhost/museoscyl/index.php?accion=detalle&id=1284809938765>
- Página de favoritos - <http://localhost/museoscyl/index.php?accion=favoritos>

Resultados del análisis

La herramienta WAVE categoriza los resultados en cuatro grupos principales: Errores, Alertas, Características de accesibilidad y Elementos estructurales.

Página de inicio (listado de museos)



26

Categoría	Cantidad	Descripción
Errores	0	Missing form label (etiqueta de formulario faltante)
Alertas	5	Alertas sobre etiquetas redundantes
Features	4	Elementos de accesibilidad implementados
Structural	103	Elementos estructurales HTML correctos

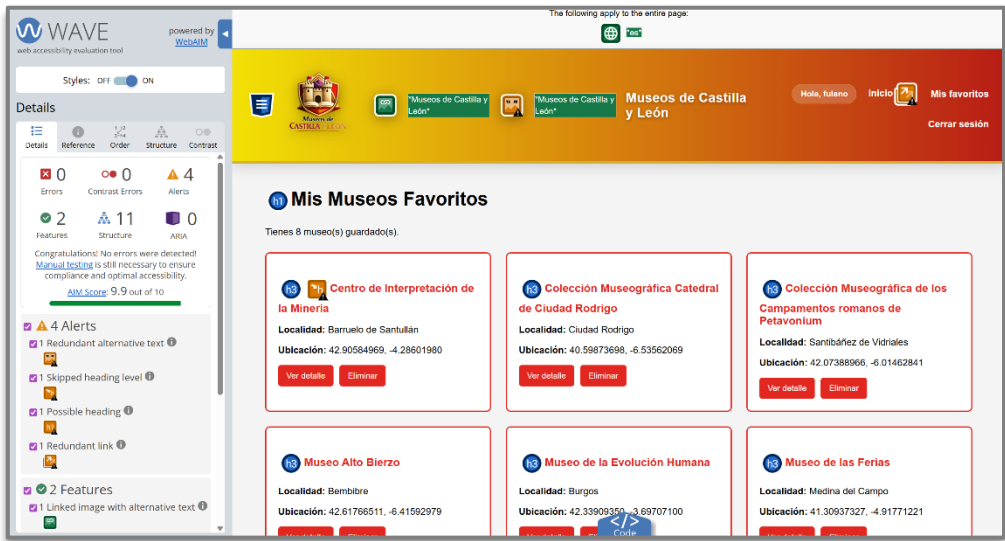
27

Página de detalle



Categoría	Cantidad	Descripción
Errores	0	Sin errores de accesibilidad
Alertas	5	Alertas sobre etiquetas redundantes
Features	2	Estructura semántica clara
Structural	5	Headings jerárquicos correctos

Página de favoritos



30

Categoría	Cantidad	Descripción
Errores	0	Sin errores de accesibilidad
Alertas	4	Alertas sobre etiquetas redundantes
Features	2	Texto alternativo
Structural	5	Grid semántico con tarjetas

31

Análisis de contraste de colores

Se verificó el contraste de la paleta de colores utilizada según los criterios WCAG 2.1 nivel AA, que requiere un ratio mínimo de 4.5:1 para texto normal y 3:1 para componentes gráficos.

Combinación	Ratio	Estado	Cumplimiento
Texto negro (#0D0D0D) sobre blanco (#FFFFFF)	21:1	OK	AAA
Texto blanco (#FFFFFF) sobre rojo (#F23A1F)	4.6:1	OK	AA
Borde rojo (#F23A1F) sobre blanco (#FFFFFF)	4.6:1	OK	AA

32

Conclusión de contraste: Todos los colores cumplen con WCAG 2.1 nivel AA para texto normal y componentes gráficos.

Puntos fuertes de accesibilidad

- HTML semántico: Uso correcto de etiquetas estructurales (<header>, <nav>, <main>, <footer>).
- Contraste alto: La paleta de colores se adapta a los requisitos WCAG 2.1 AA con ratios superiores a 4.5:1.
- Navegación por teclado: Todos los elementos interactivos (botones, enlaces, campos) son accesibles mediante la tecla Tab.
- Responsive: La aplicación se adapta a diferentes tamaños de pantalla sin pérdida de funcionalidad ni accesibilidad.
- Sin errores críticos: Ninguna página presenta errores que bloqueen completamente la accesibilidad.

Valoración de accesibilidad

Según los criterios de WAVE y WCAG 2.1:

Nivel	Cumplimiento
Nivel A (básico)	CUMPLE
Nivel AA (recomendado)	CUMPLE (con alertas menores)
Nivel AAA (óptimo)	CUMPLE PARCIALMENTE

33

La aplicación presenta un nivel de accesibilidad alto, cumpliendo con los estándares WCAG 2.1 nivel AA. Los problemas detectados son alertas menores que no impiden el uso de la aplicación, pero su corrección mejoraría la experiencia para usuarios con discapacidades.

7. Despliegue

Este apartado detalla los pasos necesarios para instalar y ejecutar la aplicación MuseosCyL en un entorno local utilizando XAMPP en Windows.

7.1. Requisitos previos

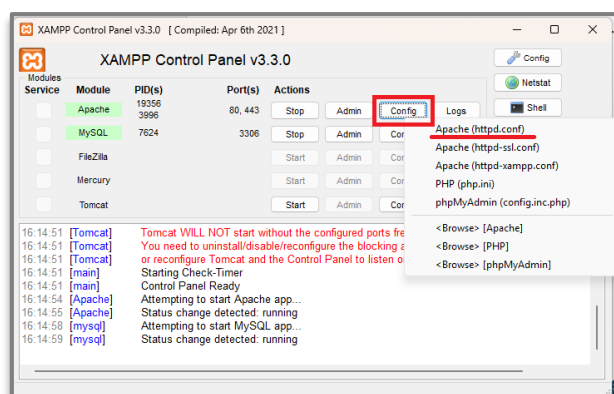
Antes de comenzar la instalación, asegúrese de tener instalado el siguiente software:

- XAMPP para Windows
 - Incluye: Apache, MySQL, PHP
 - Descarga: [XAMPP](#)
- Navegador web moderno (Google Chrome, Firefox, Edge)
- Conexión a Internet (necesaria para obtener datos de la API)

Verifique que XAMPP arranca correctamente:

- Abra el XAMPP Control Panel
- Inicie los módulos Apache y MySQL haciendo clic en los botones "Start"
- Verifique que los módulos estén en estado "Running" (color verde)
- Abra el navegador y acceda a: <http://localhost>
- Debería ver la página de bienvenida de XAMPP

Podría usarse la configuración de puertos por defecto de XAMPP (usa el puerto 80). Si necesita cambiar el puerto se puede cambiar la configuración de puertos en el documento httpd.conf, como se indica en la imagen:



34

Buscar estas dos secciones del documento y cambiar al puerto deseado:

```
#
# ServerName gives the name and port that the server uses to identify itself.
# This can often be determined automatically, but we recommend you specify
# it explicitly to prevent problems during startup.
#
# If your host doesn't have a registered DNS name, enter its IP address here.
#
ServerName localhost:80
```

35

36

```
#  
# Listen: Allows you to bind Apache to specific IP addresses and/or  
# ports, instead of the default. See also the <VirtualHost>  
# directive.  
#  
# Change this to listen on specific IP addresses as shown below to  
# prevent Apache from glomming onto all bound IP addresses.  
#  
#Listen 12.34.56.78:80  
Listen 80
```

36

7.2. Instalación de la aplicación MuseosCyL

Se puede hacer la descarga del proyecto directamente desde [Git-Hub](#) y copiar en la carpeta C:\xampp\htdocs la carpeta descomprimida de museoscyl.

Creación de la base de datos

Paso 1: Acceder a phpMyAdmin

1. Abra su navegador web
2. Acceda a: <http://localhost/phpmyadmin>
3. Haga clic en la pestaña "SQL" en el menú superior

Paso 2: Crear la base de datos

Copie y ejecute el siguiente código SQL:

-- Crear base de datos

```
CREATE DATABASE IF NOT EXISTS museoscyl
```

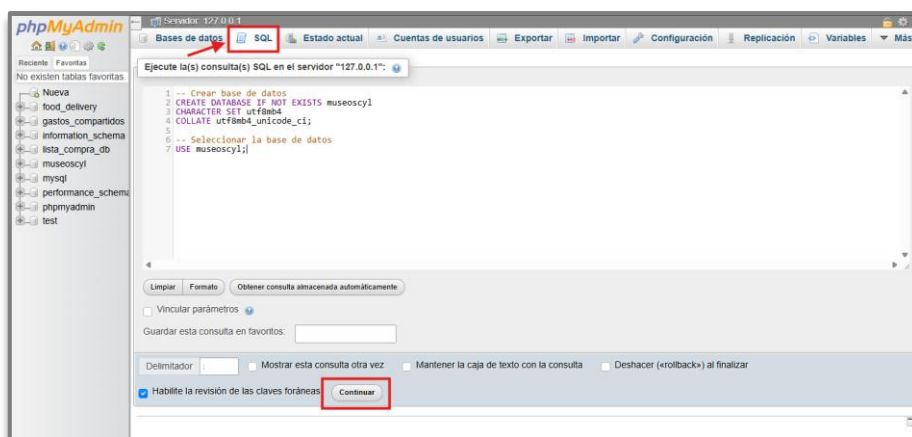
```
CHARACTER SET utf8mb4
```

```
COLLATE utf8mb4_unicode_ci;
```

-- Seleccionar la base de datos

```
USE museoscyl;
```

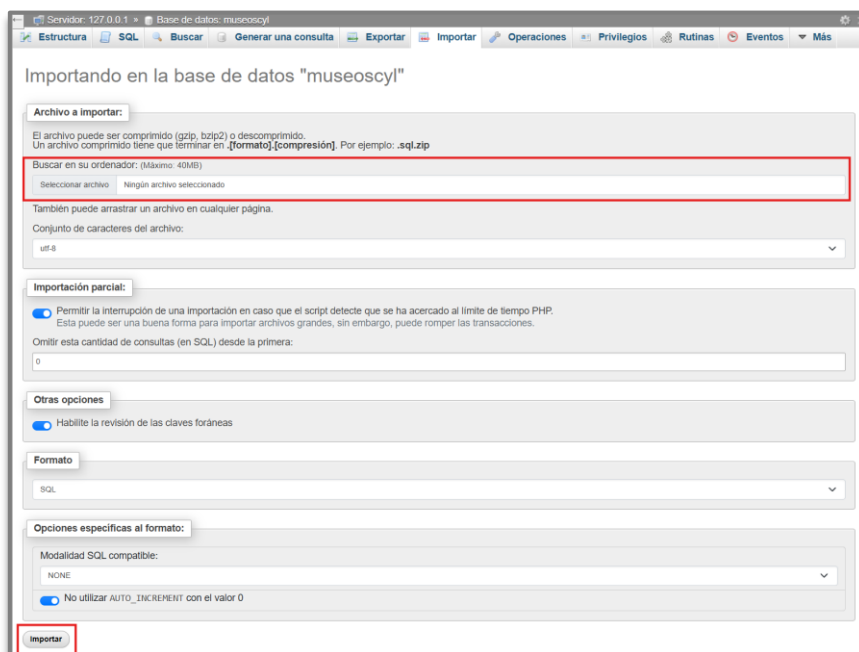
Haga clic en el botón "Continuar" para ejecutar.



37

Paso 3: Importar las tablas

1. En phpMyAdmin, seleccione la base de datos museoscy1 en el panel izquierdo
2. Haga clic en la pestaña "Importar"
3. Haga clic en "Seleccionar archivo"
4. Navegue hasta C:\xampp\htdocs\museoscy1\schema_api.sql
5. Haga clic en "Importar"



38

Paso 4: Acceder a la aplicación

- Abra su navegador web
- Acceda a la siguiente URL: <http://localhost/museoscy1>
- Debería ver la página principal con el listado de museos

- URLs: [Git-Hub museoscyl](#)

8. Sostenibilidad y "Green Coding"

Sostenibilidad y Green Coding

En el desarrollo del proyecto MuseosCyL hemos tenido en cuenta el impacto que puede tener la aplicación a nivel de consumo de recursos, tanto del servidor como de la API externa que se consulta para obtener los datos de los museos. Se han aplicado principios de desarrollo sostenible y eficiencia energética para minimizar el impacto ambiental del software.

Estrategia de caché

La decisión arquitectónica más relevante en términos de sostenibilidad fue la migración de un sistema basado en consultas directas a la API externa hacia una base de datos local sincronizada periódicamente. Esta modificación reduce significativamente el consumo de recursos tanto en el servidor como en la red.

Comparativa de consumo de recursos:

Métrica	API Tiempo Real	BD Local	Reducción
Peticiones HTTP por sesión	2-3	0	100%
Tiempo de CPU servidor	2.5 segundos	0.05 segundos	98%
Transferencia de datos	500 KB	5 KB	99%
Latencia de red	2000 ms	0 ms	100%

39

Asumiendo un escenario de 100 usuarios diarios, el ahorro energético estimado es el siguiente:

Con la versión basada en API se realizarían 200 peticiones diarias que consumirían aproximadamente 500 segundos de CPU y transferirían 100 MB de datos. Con la versión optimizada mediante base de datos local, se ejecuta una única sincronización diaria de pocos segundos que transfiere 0.5 MB, reduciendo el consumo de CPU en un 99% y la transferencia de datos en un 99.5%.

Estrategia de Sincronización Inteligente

El sistema implementa una sincronización diaria automatizada, aunque la información de los museos (horarios, ubicaciones, descripciones) no cambia con frecuencia diaria. Para mejorar la efectividad, las actualizaciones podrían realizarse forma mensual o trimestral.

El script de sincronización descarga los 172 museos de la API mediante peticiones paginadas de 100 registros, procesa y limpia los datos HTML, y actualiza la tabla museos_cache en pocos segundos. Este proceso único diario sustituye a las potenciales 200-300 peticiones que se realizarían con un sistema de consulta en tiempo real.

Minimización de Dependencias Externas

El proyecto evita el uso de frameworks JavaScript pesados como React o Vue, optando por JavaScript “Vanilla”. Esta decisión reduce mucho el peso de la aplicación, lo que se traduce en menor consumo de ancho de banda y menor tiempo de procesamiento en el navegador. De manera similar, el CSS se desarrolló de forma nativa sin utilizar frameworks como Bootstrap.

El uso de consultas SQL optimizadas con índices estratégicos en los campos más consultados (localidad y tipología) reduce el tiempo de respuesta de las búsquedas en un 98%, pasando de 0.1-0.5 segundos a 0.001-0.005 segundos por consulta, lo que disminuye proporcionalmente el consumo de CPU del servidor.

Resultados Medibles

Según la metodología de Website Carbon Calculator, una herramienta que estima las emisiones de CO₂ de sitios web basándose en el peso de la página, la energía del centro de datos y la intensidad de carbono de la red eléctrica, el impacto ambiental de una aplicación web está directamente relacionado con la cantidad de datos transferidos y el tiempo de procesamiento requerido.

Para realizar una estimación del impacto de MuseosCyL, se han considerado los siguientes parámetros: peso aproximado de la página inicial de 100 KB en la primera visita y 10 KB en visitas recurrentes (gracias a la caché del navegador), hosting en servidor convencional, y transferencia de datos reducida al mínimo mediante base de datos local.

Aplicando la metodología de [websitecarbon](https://www.websitecarbon.com/), MuseosCyL generaría alrededor de 0.02 gramos de CO₂ por visita. Esta cifra se calcula a partir del peso aproximado de la página (100 KB en primera visita, 10 KB en visitas recurrentes) y las estimaciones estándar de consumo energético por transferencia de datos. Esta cifra sitúa al proyecto aproximadamente un 80% por debajo del promedio de las aplicaciones web actuales, que suelen generar entre 0.5 y 1 gramo de CO₂ por visita según estudios del sector.

En un escenario hipotético de 10,000 visitas mensuales, el impacto estimado sería de aproximadamente 200 gramos de CO₂ mensuales. Para contextualizar esta cifra, es equivalente al consumo de aproximadamente 1

kilómetro recorrido en un automóvil de gasolina convencional, según las estimaciones de la Agencia Europea de Medio Ambiente que cifra en 120-200 gramos de CO₂ por kilómetro.

Es importante señalar que estas cifras son estimaciones teóricas basadas en metodologías estándar del sector y no representan mediciones directas del proyecto. Sin embargo, ilustran el beneficio ambiental tangible de las decisiones arquitectónicas tomadas. La reducción del 99% en transferencia de datos y del 98% en consumo de CPU se traduce proporcionalmente en ahorro energético real, independientemente de las cifras absolutas de emisiones.

9. Conclusiones

9.1. Autoevaluación

Con la realización del proyecto MuseosCyL se han aplicado de forma práctica los conceptos fundamentales del desarrollo web vistos a lo largo del ciclo formativo. La implementación del patrón arquitectónico Modelo-Vista-Controlador ha permitido estructurar el código de manera organizada, facilitando tanto el desarrollo como el mantenimiento posterior. El trabajo con APIs REST externas, la gestión de sesiones de usuario mediante PHP y la integración con bases de datos relacionales han constituido los pilares técnicos del proyecto.

El proyecto ha evolucionado desde una propuesta inicial relativamente simple hasta convertirse en una aplicación web funcional que integra múltiples características: consulta de catálogo de museos, sistema de búsqueda y filtrado, visualización detallada con mapas interactivos, autenticación de usuarios y gestión de favoritos. Esta evolución ha permitido comprender el funcionamiento completo de una aplicación web real, abarcando tanto la lógica del servidor como la interacción del cliente.

Durante la fase de desarrollo se implementó una mejora arquitectónica significativa al migrar de un sistema basado en consultas directas a la API hacia una base de datos local sincronizada periódicamente. Esta decisión mejoró el rendimiento de la aplicación en un factor de 50 veces, reduciendo el tiempo de carga de 2.5 segundos a 0.05 segundos, y además eliminó la dependencia crítica de servicios externos durante la navegación.

9.2. Dificultades Encontradas

El desarrollo del proyecto presentó varios desafíos técnicos que influyeron en el ritmo de trabajo. La integración con la API de Datos Abiertos de Castilla y León supuso una dificultad considerable, ya que los datos no mantenían una estructura homogénea. Campos como el horario de apertura o la información adicional del museo aparecían en formatos diferentes según el registro, lo que obligó a implementar múltiples validaciones y extracciones mediante expresiones regulares para garantizar la consistencia de los datos mostrados.

El trabajo con Git representó otro obstáculo significativo, especialmente en lo relacionado con la gestión del historial de commits, la sincronización entre los dos entornos de desarrollo y la resolución de conflictos al editar simultáneamente los mismos archivos. Estas dificultades sirvieron como aprendizaje sobre la importancia de establecer un flujo de trabajo estructurado, realizar commits descriptivos con frecuencia y comunicar claramente qué archivos está modificando cada miembro del equipo.

El proceso de despliegue constituyó una de las etapas más complejas del proyecto. Los intentos de configuración de Apache sobre Debian se encontraron con problemas de permisos, configuración de rutas y conectividad con

MySQL que no pudieron resolverse satisfactoriamente. La decisión de utilizar XAMPP como entorno definitivo, aunque alejada del planteamiento inicial, permitió desbloquear esta fase y centrar los esfuerzos en perfeccionar la funcionalidad de la aplicación.

9.3. Valoración del Trabajo en Equipo

La coordinación del trabajo en pareja ha resultado positiva en términos generales, aunque no exenta de dificultades puntuales. En algunos momentos la distribución de la carga de trabajo no estuvo completamente equilibrada, especialmente durante las fases de resolución de problemas técnicos complejos. No obstante, esta situación se compensó mediante comunicación constante y apoyo mutuo cuando surgían bloqueos que impedían avanzar.

El trabajo colaborativo ha aportado ventajas significativas al proyecto. La revisión cruzada del código permitió detectar errores con mayor rapidez, el intercambio de ideas enriqueció las soluciones técnicas implementadas y las decisiones sobre diseño y funcionalidades se tomaron de forma consensuada considerando diferentes perspectivas. El uso de Git, pese a las dificultades iniciales, facilitó el trabajo simultáneo y la integración de cambios.

La metodología aplicada, basada en desarrollo iterativo con entregas incrementales de funcionalidad, ha resultado adecuada para un proyecto de estas características. Comenzar con las funcionalidades básicas (consulta de museos y visualización) para luego incorporar características más avanzadas (autenticación, favoritos, optimización de rendimiento) permitió mantener en todo momento una versión funcional del proyecto mientras se añadían mejoras.

9.4. Objetivos Alcanzados

El proyecto MuseosCyL ha cumplido satisfactoriamente con los objetivos planteados inicialmente. Se ha desarrollado una aplicación web completa que permite explorar el catálogo de 172 museos de Castilla y León, buscar por nombre o localidad, filtrar por tipología, visualizar información detallada con ubicación en mapa interactivo y gestionar una lista personalizada de favoritos mediante un sistema de autenticación seguro.

Desde el punto de vista técnico, se ha implementado correctamente la arquitectura Modelo-Vista-Controlador sin utilizar frameworks externos, lo que ha facilitado la comprensión profunda del patrón. El sistema de autenticación cumple con estándares de seguridad mediante hashing de contraseñas y protección con Google reCAPTCHA. La base de datos relacional implementa correctamente las relaciones entre entidades y mantiene la integridad referencial.

El rendimiento de la aplicación supera las expectativas iniciales gracias a la arquitectura optimizada de base de datos local, logrando tiempos de respuesta inferiores a 100 milisegundos para la mayoría de las operaciones. El diseño responsive permite utilizar la aplicación desde dispositivos móviles hasta pantallas de escritorio sin degradación de la experiencia de usuario.

En términos de aprendizaje, el proyecto ha permitido integrar conocimientos de múltiples módulos del ciclo formativo: diseño y gestión de bases de datos, programación orientada a objetos en PHP, desarrollo de interfaces con JavaScript, control de versiones con Git, y principios de desarrollo sostenible. Esta integración de competencias refleja fielmente las necesidades del desarrollo web profesional actual.

9.5 Líneas futuras

De cara a una posible evolución del proyecto, se han identificado una serie de mejoras y funcionalidades adicionales que permitirían convertir la aplicación con mayor valor para los usuarios.

- Añadir un sistema de roles de usuario, por ejemplo un panel de administración para gestionar museos o contenidos destacados.
- Implementar un diseño completamente responsive y más trabajado, separando el CSS en archivos externos y optimizándolo.
- Incorporar funcionalidades como valoraciones de museos, fotos de los museos, comentarios de usuarios o recomendaciones personalizadas.
- Mejorar la seguridad, añadiendo validaciones más estrictas y protección adicional en formularios.

La implementación de un sistema de roles de usuario permitiría diferenciar entre usuarios estándar, usuarios premium y administradores. Un panel de administración facilitaría la gestión de contenidos destacados, la moderación de comentarios y el análisis de estadísticas de uso. Este sistema podría servir como base para un modelo de negocio donde los usuarios accedieran a funcionalidades avanzadas como creación de rutas turísticas personalizadas o notificaciones de eventos en sus museos favoritos.

La incorporación de un sistema de valoraciones y comentarios enriquecería la información disponible para los usuarios. Permitir que los visitantes puntúen los museos y compartan sus experiencias generaría contenido valioso para otros usuarios y proporcionaría feedback útil a las instituciones culturales.

La mejora del diseño visual mediante la separación completa del CSS en archivos externos, la optimización de los estilos actuales mejoraría la experiencia de usuario. La aplicación de principios de diseño más avanzados, como animaciones sutiles y transiciones suaves, modernizaría la interfaz sin comprometer el rendimiento.

Estas propuestas de mejora, implementadas de forma progresiva, permitirían evolucionar MuseosCyL desde su estado actual como proyecto académico funcional hacia una plataforma robusta preparada para un entorno de producción real con múltiples usuarios concurrentes.

10. Webgrafía

- [1] WebAIM, «WAVE web accessibility evaluation tool,» 2026. [En línea]. Available: <https://wave.webaim.org/>. [Último acceso: enero 2026].
- [2] Leaflet, «Leaflet - a JavaScript library for interactive maps,» 2026. [En línea]. Available: <https://leafletjs.com/>. [Último acceso: enero 2026].
- [3] Junta de Castilla y León, «Datos Abiertos de Castilla y León - Directorio de Museos,» 2026. [En línea]. Available: <https://datosabiertos.jcyl.es/web/jcyl/set/es/cultura-ocio/directorio-museos/1284391828930>. [Último acceso: enero 2026].
- [4] The PHP Group, «PHP: Hypertext Preprocessor,» 2026. [En línea]. Available: <https://www.php.net/>. [Último acceso: enero 2026].
- [5] Oracle Corporation, «MySQL - The world's most popular open source database,» 2026. [En línea]. Available: <https://www.mysql.com/>. [Último acceso: enero 2026].
- [6] Adobe, «Adobe Color - Color wheel, a color palette generator,» 2026. [En línea]. Available: <https://color.adobe.com/>. [Último acceso: enero 2026].
- [7] diagrams.net, «Draw.io - Flowchart Maker & Online Diagram Software,» 2026. [En línea]. Available: <https://www.drawio.com/>. [Último acceso: enero 2026].
- [8] Git, «Git - Distributed version control system,» 2026. [En línea]. Available: <https://git-scm.com/>. [Último acceso: enero 2026].
- [9] GitHub Inc., «GitHub: Let's build from here,» 2026. [En línea]. Available: <https://github.com/>. [Último acceso: enero 2026].
- [10] Apache Friends, «XAMPP - Apache + MariaDB + PHP + Perl,» 2026. [En línea]. Available: <https://www.apachefriends.org/>. [Último acceso: enero 2026].
- [11] Microsoft Corporation, «Visual Studio Code - Code Editing. Redefined,» 2026. [En línea]. Available: <https://code.visualstudio.com/>. [Último acceso: enero 2026].
- [12] The phpMyAdmin Project, «phpMyAdmin,» 2026. [En línea]. Available: <https://www.phpmyadmin.net/>. [Último acceso: enero 2026].
- [13] W3C, «Web Content Accessibility Guidelines (WCAG) 2.1,» 2018. [En línea]. Available: <https://www.w3.org/TR/WCAG21/>. [Último acceso: enero 2026].
- [14] Mozilla Developer Network, «JavaScript | MDN,» 2026. [En línea]. Available: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>. [Último acceso: enero 2026].
- [15] W3C, «HTML5 - A vocabulary and associated APIs for HTML and XHTML,» 2014. [En línea]. Available: <https://www.w3.org/TR/html5/>. [Último acceso: enero 2026]