# SPOTMAP

**Plataforma web para descubrir y compartir lugares fotográficos**

**Anteproyecto de Fin de Grado Superior en Desarrollo de Aplicaciones Web**

**Alumno:** Antonio Valero Maldonado  
**Curso académico:** 2025

**Índice**

1. Título
2. Introducción
   * 2.1 Contexto
   * 2.2 Situación actual
   * 2.3 Fortalezas del proyecto
   * 2.4 Vulnerabilidades del proyecto
3. Objetivo e innovaciones
   * 3.1 Objetivo general
   * 3.2 Objetivos específicos
   * 3.3 Innovaciones
4. Alcance y requisitos
5. Diseño funcional y técnico
   * 5.1 Arquitectura propuesta
   * 5.2 Modelo de datos
   * 5.3 Casos de uso principales
   * 5.4 Interfaces de usuario
6. Medios de hardware y software
7. Seguridad, privacidad y aspectos legales
8. Planificación
9. Pruebas, métricas y evaluación
10. Futuro y escalabilidad
11. Bibliografía

**2. Introducción**

**2.1 Contexto**

Las redes sociales como Instagram o TikTok han cambiado la forma en la que los usuarios descubren su entorno. Hoy en día, encontrar “spots” fotográficos lugares con encanto o buenas vistas para sacar fotos es parte de la experiencia de viaje y ocio. Sin embargo, no existe una plataforma centralizada y sencilla que permita a los usuarios localizar y compartir estos lugares.

**2.2 Situación actual**

Existen soluciones parciales:

* **Google Maps**: muy completo en ubicaciones, pero no especializado en fotografía.
* **Pinterest**: permite buscar inspiración visual, pero sin geolocalización precisa.
* **Aplicaciones turísticas**: suelen ser locales y poco actualizadas.

La falta de una plataforma especializada en spots fotográficos deja un espacio claro para un proyecto innovador.

**2.3 Fortalezas del proyecto**

1. Plataforma dedicada exclusivamente a spots fotográficos.
2. Modelo colaborativo que crece con aportaciones de la comunidad.
3. Base técnica escalable y adaptable a móviles.
4. Preparado para integraciones futuras con IA.

**2.4 Vulnerabilidades del proyecto**

1. Riesgo de contenido inapropiado o mal categorizado.
2. Dificultad inicial de atraer usuarios (masa crítica).
3. Posibles conflictos legales sobre privacidad o derechos de imagen.
4. Riesgo de saturación en spots populares.

**3. Objetivo e innovaciones**

**3.1 Objetivo general**

Desarrollar una aplicación web colaborativa que permita descubrir, compartir y valorar lugares fotográficos, con un diseño atractivo y fácil de usar, construida sobre una arquitectura flexible y escalable.

**3.2 Objetivos específicos**

* Sistema de autenticación seguro con distintos roles (usuario, moderador, administrador).
* Mapa interactivo con búsqueda avanzada por cercanía, categorías y popularidad.
* Posibilidad de añadir spots con fotos, descripción, etiquetas y ubicación precisa.
* Filtros útiles como mejor hora para la foto, dificultad de acceso o temporada recomendada.
* Gamificación mediante puntuaciones y logros.
* Panel de administración para moderar contenido y gestionar estadísticas.
* Preparar la base técnica para futuras integraciones de inteligencia artificial.

**3.3 Innovaciones**

* Incluir metadatos relevantes para fotografía (orientación solar, altitud, condiciones de luz).
* Clasificación semántica de los spots (playa, mirador, arquitectura, graffiti, etc.).
* Compartir spots o rutas directamente en redes sociales.
* Sistema de reporte de problemas relacionados con masificación o deterioro ambiental.

**4. Alcance y requisitos**

El proyecto será una **aplicación web responsiva** (PC y móvil).

**MVP (mínima versión funcional):**

* Usuarios: registro/login, añadir spots, búsqueda con filtros, valorar y comentar.
* Administradores: moderar spots, gestionar usuarios y obtener estadísticas básicas.

**5. Diseño funcional y técnico**

**5.1 Arquitectura propuesta**

La aplicación seguirá un modelo clásico de **tres capas**:

1. **Frontend**: desarrollado en JavaScript con librerías modernas, será la parte visible para los usuarios (mapa, formularios, perfiles).
2. **Backend**: desarrollado en **PHP (con frameworks como Laravel o Symfony)**, que se encargará de la lógica del sistema, la gestión de usuarios y la comunicación con la base de datos.
3. **Base de datos**: **MongoDB**, aprovechando sus índices geoespaciales para búsquedas por ubicación.

**Servicios complementarios:**

* Almacenamiento en la nube (ej. AWS S3) para fotos.
* ElasticSearch para búsquedas avanzadas de spots.
* Procesamiento de imágenes (miniaturas, lectura de metadatos EXIF).
* Colas de tareas (ej. RabbitMQ) para notificaciones y análisis de segundo plano.

**5.2 Modelo de datos simplificado**

* **Usuario** (id, nombre, email, hash\_password, rol, reputación).
* **Spot** (id, usuario\_id, título, descripción, coordenadas, etiquetas[], fotos[], estado, fecha).
* **Comentario** (id, spot\_id, usuario\_id, texto, fecha).
* **Valoración** (id, spot\_id, usuario\_id, puntuación).
* **Reporte** (id, spot\_id, usuario\_id, tipo, descripción, estado).

**5.3 Casos de uso principales**

1. Buscar spots cercanos filtrados por categoría.
2. Publicar un spot con fotos, ubicación y descripción.
3. Comentar y valorar spots de otros usuarios.
4. Moderar un spot reportado.

**5.4 Interfaces clave**

* Página principal con mapa y buscador.
* Ficha detallada del spot.
* Formulario de alta de spots.
* Perfil de usuario.
* Panel de moderación.

**6. Medios de hardware y software**

**Hardware necesario:**

* Servidor cloud escalable (ej. AWS, Azure, Google Cloud).
* PC de desarrollo (mín. i5, 8GB RAM, SSD).
* Dispositivos móviles para pruebas (Android e iOS).

**Software necesario:**

* **Frontend**: JavaScript + librerías UI.
* **Backend**: PHP (Laravel/Symfony).
* **Base de datos**: MongoDB.
* **Control de versiones**: GitHub/GitLab.
* **Gestión de proyecto**: Trello.
* **Diseño**: Figma, Canva.
* **Pruebas**: PHPUnit (backend) + Cypress (frontend).

**7. Seguridad, privacidad y aspectos legales**

* Cumplimiento de **GDPR**: derecho a exportar o borrar datos.
* Declaración de derechos de imagen al subir fotos.
* Hash de contraseñas con bcrypt.
* Tokens JWT para sesiones seguras.
* Políticas claras de moderación y protección de datos.

**8. Planificación**

**Metodología:** Scrum con sprints de 2 semanas.

| **Sprint** | **Duración** | **Objetivos** | **Entregables** |
| --- | --- | --- | --- |
| 1 | 2 semanas | Requisitos y wireframes | Documento + prototipos |
| 2–3 | 4 semanas | Backend (auth, API spots) | API funcional + tests |
| 4–5 | 4 semanas | Frontend (mapa, fichas) | MVP integrado |
| 6 | 2 semanas | Moderación y valoraciones | Panel moderador |
| 7 | 2 semanas | Despliegue y documentación | App online + manual |

Duración estimada: **16 semanas**.

**9. Pruebas, métricas y evaluación**

* **Pruebas**: unitarias, integración, E2E y de carga.
* **Métricas**: usuarios activos, spots creados, tiempo de aprobación, retención.
* **Éxito del MVP**: 500 usuarios activos y 1000 spots en 3 meses.

**10. Futuro y escalabilidad**

* Migración progresiva a microservicios si aumenta la carga.
* CDNs para optimizar imágenes.
* Aplicación móvil nativa (Flutter/React Native).
* Recomendaciones personalizadas por IA.
* Integración con oficinas de turismo y fotógrafos locales.

**11. Bibliografía**

* Sommerville, I. (2011). *Ingeniería del Software*. Pearson.
* Pressman, R. (2010). *Software Engineering: A Practitioner’s Approach*. McGraw-Hill.
* Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). *Design Patterns*. Addison-Wesley.
* LeafletJS Documentation: <https://leafletjs.com/>
* MongoDB Documentation: <https://www.mongodb.com/docs/>
* PHP Manual: <https://www.php.net/manual/es/>
* Symfony Documentation: https://symfony.com/doc/current/index.html
* Scrum Guides: <https://scrumguides.org/>
* GDPR - Regulation (EU) 2016/679