SpotMap

Desarrollo de aplicación WEB

CURSO 2025/26

Alumno/a

Antonio Valero Maldonado

Tutor/a:

XXXXXX XXXXXX XXXXX

UNENDO

CICLO FORMATIVO DE GRADO SUPERIOR

EN DESARROLLO DE APLICACIONES WEB

Índice

1. Descripción general del proyecto

1.1 Introducción

SpotMap es una plataforma web colaborativa cuyo objetivo principal es centralizar, organizar y facilitar el acceso a lugares fotográficos —miradores, espacios naturales, rincones urbanos y escenarios visuales— mediante un sistema de geolocalización interactivo. A diferencia de redes sociales como Instagram o TikTok, donde el contenido está disperso y no clasificado para este propósito, SpotMap se centra exclusivamente en recopilar spots de interés fotográfico con información estructurada, validada y enriquecida por la comunidad.

El proyecto surge como respuesta a la creciente demanda de herramientas que permitan a fotógrafos, viajeros y creadores de contenido localizar fácilmente escenarios visuales de calidad. SpotMap ofrecerá una experiencia completa: desde descubrir localizaciones mediante un mapa dinámico hasta consultar descripciones detalladas, horarios recomendados, consejos de fotografía y opiniones de otros usuarios.

La plataforma está dirigida a:

- Fotógrafos profesionales y aficionados.
- Creadores de contenido e influencers.
- Turistas que buscan experiencias menos convencionales.
- Instituciones turísticas interesadas en difundir su entorno mediante contenido visual de calidad.

SpotMap pretende convertirse en un repositorio global, colaborativo y moderado de spots fotográficos, fomentando tanto la participación comunitaria como la calidad del contenido.

1.2 Alcance del proyecto

El sistema cubrirá las siguientes funcionalidades:

1. Gestión de usuarios: registro, inicio de sesión, cierre de sesión y recuperación de contraseña.
2. Gestión de roles: usuarios registrados, moderadores y administradores.
3. Gestión de spots: creación, edición, eliminación y visualización de spots con imágenes, etiquetas, ubicación y descripciones.
4. Búsqueda avanzada: filtros por categoría, distancia, popularidad, etiqueta u horario recomendado.
5. Interacción social: valoraciones, comentarios y marcadores de favoritos.
6. Mapa interactivo: visualización de coordenadas con marcadores dinámicos.
7. Moderación: revisión de contenido inapropiado, duplicado o reportado.
8. Panel administrativo: gestión de usuarios, estadísticas de uso, reportes y métricas globales.

1.3 Justificación y análisis de la realidad

En la actualidad, la búsqueda de localizaciones fotográficas depende principalmente de publicaciones en redes sociales, blogs personales o recomendaciones informales. Este enfoque presenta numerosos problemas: la información no está organizada, los lugares no siempre cuentan con coordenadas precisas, la calidad del contenido es variable y, en general, no existe una plataforma única que permita localizar spots de manera eficiente.

Además, muchos fotógrafos y creadores invierten tiempo significativo navegando entre múltiples fuentes para encontrar lugares adecuados. Las instituciones turísticas también carecen de herramientas específicas para mostrar el potencial visual de su entorno.

SpotMap responde a estas necesidades mediante:

- Un sistema centralizado de localización fotográfica basado en geolocalización.
- Contenido generado por la comunidad con mecanismos de validación y moderación.
- Una interfaz intuitiva que facilita la exploración y el descubrimiento de lugares.
- Beneficios para organismos locales que deseen promocionar su entorno de forma visual y ordenada.

Desde un punto de vista formativo, el proyecto integra desarrollo web, bases de datos geoespaciales, diseño de APIs, seguridad, control de versiones y gestión de proyectos.

1.4 Marco legal

El proyecto cumple con el Reglamento General de Protección de Datos (GDPR - UE 2016/679). El usuario podrá:

- Modificar y eliminar sus datos personales.
- Gestionar derechos de imagen en el contenido subido.
- Aceptar los términos y condiciones antes del registro.

La plataforma incluirá:

- Política de privacidad detallada.
- Aviso legal y condiciones de uso.
- Normas de publicación y código de conducta.

Se prestará especial atención al uso responsable del contenido, especialmente en lo relativo a derechos de autor, privacidad y protección de menores.

1.5 Marco teórico

SpotMap se basa en tecnologías modernas de desarrollo web:

- Arquitectura cliente-servidor: comunicación mediante API REST.
- Bases de datos relacional (PostgreSQL via Supabase): optimizada para consultas complejas y datos geoespaciales.

- Backend con PHP 8.2: framework robusto para desarrollo de APIs, autenticación y seguridad.
- Frontend en JavaScript (HTML/CSS/Vanilla JS): creación de interfaces responsivas.
- Diseño responsivo: adaptación a dispositivos móviles, tablets y sobremesa.
- UX/UI: priorización de la usabilidad, accesibilidad y claridad.
- Metodología Ágil: organización del trabajo en sprints iterativos.

1.6 Temporización

Duración total: 16 semanas divididas en 8 sprints.

Sprint	Duración	Objetivos
1	1 semana	Análisis inicial, estudio de requisitos, investigación de mercado.
2	1 semana	Diseño del sistema, arquitectura, esquemas de BD, wireframes.
3	2 semanas	Desarrollo de la API base, autenticación, gestión de usuarios y roles.
4	2 semanas	CRUD de spots, subida de imágenes y validaciones.
5	2 semanas	Mapa interactivo, integración con API de geolocalización.
6	2 semanas	Comentarios, valoraciones, sistema de búsqueda avanzada.
7	2 semanas	Moderación, panel administrativo, estadísticas básicas.
8	2 semanas	Pruebas, documentación, despliegue final y optimización.

2. Estudio de la viabilidad del sistema

2.1 Estudio de la viabilidad del sistema (DAFO)

Fortalezas

- Proyecto original con un alto valor turístico, social y cultural.
- Escalable tanto a nivel nacional como internacional debido al carácter universal de la fotografía.
- Uso de tecnologías libres y open source, lo que reduce los costes de desarrollo y mantenimiento.
- Posibilidad de generar una comunidad activa que aporte contenido de forma continua.
- Plataforma especializada en spots fotográficos, ofreciendo un valor que redes generalistas no cubren.
- Capacidad de integrarse con herramientas de geolocalización y APIs modernas.

Debilidades

- Necesidad de alcanzar una masa crítica de usuarios para que la plataforma resulte realmente útil.
- Dependencia directa de la calidad, veracidad y frecuencia del contenido generado por los usuarios.
- Requerimiento constante de moderación para evitar contenido inapropiado, duplicado o falso.
- Monetización poco clara en las primeras fases del proyecto.
- Costes potenciales por almacenamiento de imágenes en servidores en la nube.
- Riesgo de abandono del proyecto si no se mantiene una comunidad activa.

Oportunidades

- Integración con redes sociales que puede aumentar significativamente la visibilidad y viralidad del contenido.
- Colaboración con ayuntamientos, oficinas de turismo, fotógrafos locales y empresas del sector cultural.
- Posibilidad de ofrecer servicios premium a negocios locales:
    - Promociones
    - Estadísticas geográficas
    - Spots destacados o patrocinados
- Creciente demanda de aplicaciones de geolocalización, turismo experiencial y fotografía.

- Oportunidad de implementar algoritmos de recomendación o clasificación automática mediante IA.
- Posibilidad de posicionarse como herramienta educativa en escuelas de fotografía y turismo.

Amenazas

- Competencia indirecta con plataformas altamente consolidadas como Google Maps, Instagram, TikTok o TripAdvisor.
- Riesgos legales relacionados con derechos de imagen, copyright y datos personales.
- Posibles cambios en normativas locales o europeas que afecten a la publicación de contenido geolocalizado.
- Dificultad para retener usuarios si no se ofrece una propuesta de valor diferenciada y constante.
- Problemas de seguridad informática que podrían comprometer la confianza de los usuarios.

2.2 Conclusión del Estudio de Viabilidad

El coste inicial del proyecto es reducido debido al uso de herramientas open source y servicios cloud escalables, lo que permite desarrollar y mantener la plataforma sin grandes inversiones iniciales. Esta infraestructura flexible facilita el crecimiento progresivo del sistema conforme aumente el número de usuarios y el volumen de contenido.

La rentabilidad futura del proyecto podrá provenir de diversas vías:

- Publicidad ética y no intrusiva, orientada a turismo, fotografía o actividades culturales.
- Servicios premium para usuarios avanzados y negocios locales, tales como estadísticas personalizadas, posicionamiento de spots destacados o promociones geolocalizadas.
- Acuerdos con instituciones turísticas interesadas en promocionar su entorno mediante contenido visual de calidad.

Estas fuentes de ingresos, combinadas con una comunidad activa, hacen que el proyecto sea sostenible y escalable a medio y largo plazo.

2.3 Plan de Marketing

Para lograr una difusión eficaz y consolidar la comunidad inicial del proyecto, se plantean las siguientes acciones estratégicas:

1. Presencia en redes sociales

Creación de cuentas oficiales en Instagram, TikTok y X, donde se publicarán spots destacados, fotografías inspiradoras y contenido generado por usuarios. Estas plataformas actuarán como escaparate visual del proyecto.

2. Campañas colaborativas

Colaboración con fotógrafos locales, microinfluencers y creadores de contenido que puedan aportar visibilidad a la plataforma. Estas campañas permitirán alcanzar audiencias relacionadas con la fotografía y los viajes.

3. Gamificación

Implementación de un sistema de logros, insignias y recompensas para los usuarios más activos. Esto aumentará la participación, la fidelidad y el aporte continuo de nuevos spots.

4. SEO y posicionamiento web

Optimización técnica del sitio mediante estrategias de SEO on-page y off-page, con el objetivo de aparecer en las primeras posiciones de búsquedas relacionadas con fotografía, turismo y localizaciones.

5. Colaboraciones institucionales

Establecer acuerdos con oficinas de turismo, ayuntamientos y asociaciones culturales para promocionar lugares menos conocidos y reforzar la legitimidad del proyecto.

Objetivo general del marketing

Construir una base sólida de usuarios activos y motivados que permitan un crecimiento orgánico y sostenido de la plataforma, generando contenido continuo y aumentando la visibilidad del proyecto a nivel local, nacional e internacional.

3. Descripción del entorno tecnológico

3.1 Perfiles de usuario

- **Visitante**: puede navegar, ver el mapa y consultar spots.
- **Usuario registrado**: puede crear, editar, valorar y comentar spots.
- **Moderador**: revisa contenido reportado, valida spots y gestiona incidencias.
- **Administrador**: accede al panel general, estadísticas, roles y configuración global.

3.2 Tecnologías asociadas a cada perfil

Perfil	Herramientas / Tecnologías
Visitante	Navegador web moderno; frontend en JavaScript; uso de Leaflet/OpenStreetMap para el mapa.
Usuario registrado	Interfaz full stack (HTML, CSS, JavaScript); autenticación JWT; subida de imágenes (Supabase Storage); gestión de favoritos y comentarios.
Moderador	Panel interno desarrollado en HTML/CSS/JS; comunicación con API PHP; herramientas de validación de reportes; acceso a base de datos (PostgreSQL via Supabase).
Administrador	Acceso a panel avanzado con PHP, dashboards con Chart.js; gestión de roles; supervisión del servidor cloud; estadísticas y métricas globales.

4. Especificación de requisitos

4.1 Requisitos Funcionales

- **RF1. Registrarse**: El sistema debe permitir a los usuarios crear una cuenta proporcionando correo electrónico, contraseña y nombre de usuario.
- **RF2. Iniciar sesión**: El usuario debe poder acceder a su cuenta mediante correo y contraseña o a través de proveedores externos (Google, Apple…).
- **RF3. Cerrar sesión**: El sistema permitirá al usuario desconectarse de su cuenta de forma segura en cualquier momento.
- **RF4. Crear spots fotográficos**: El usuario podrá subir la localización, una fotografía y una breve descripción del lugar.
- **RF5. Editar spots propios**: El usuario podrá modificar información de los spots que haya creado (descripción, título o imagen).
- **RF6. Eliminar spots propios**: El usuario tendrá la posibilidad de borrar spots que ya no desee mantener publicados.
- **RF7. Explorar spots en el mapa**: El usuario podrá visualizar spots geolocalizados mediante un mapa interactivo y filtrarlos según categoría, popularidad o cercanía.
- **RF8. Buscar spots**: El sistema permitirá buscar spots por nombre, palabra clave o localidad.
- **RF9. Marcar favoritos**: El usuario podrá guardar spots para acceder a ellos rápidamente desde su perfil.
- **RF10. Comentar spots**: El sistema permitirá añadir comentarios en spots ajenos para fomentar la interacción.
- **RF11. Moderar contenido**: Los administradores podrán revisar, aprobar, ocultar o eliminar contenido que incumpla las normas.
- **RF12. Reportar contenido**: Cualquier usuario podrá denunciar un spot o comentario inapropiado, indicando el motivo.
- **RF13. Gestionar usuarios**: Los administradores podrán suspender cuentas, restablecer contraseñas o detectar actividad sospechosa.
- **RF14. Generar estadísticas**: El sistema recopilará datos básicos (visitas, spots más populares, usuarios activos) para análisis interno.
- **RF15. Recibir notificaciones**: El usuario podrá recibir avisos sobre comentarios, favoritos o cambios en sus spots.

4.2 Requisitos No Funcionales

- **RNF1. Usabilidad**: La interfaz debe ser intuitiva, accesible y adaptada a usuarios sin conocimientos técnicos.

- **RNF2. Rendimiento**: El sistema debe cargar resultados de búsqueda o mapa en menos de 3 segundos bajo condiciones normales.
- **RNF3. Seguridad**: Todos los datos personales deben almacenarse cifrados y transmitirse mediante HTTPS.
- **RNF4. Escalabilidad**: La arquitectura debe soportar un incremento progresivo de usuarios sin perder rendimiento.
- **RNF5. Disponibilidad**: El sistema debe garantizar un funcionamiento del 99% mensual, salvo tareas de mantenimiento planificadas.
- **RNF6. Compatibilidad**: La aplicación debe funcionar correctamente en navegadores modernos y dispositivos móviles Android/iOS.
- **RNF7. Mantenibilidad**: El código debe estar documentado y organizado para permitir modificaciones futuras.
- **RNF8. Privacidad**: El tratamiento de datos debe cumplir con la legislación vigente (RGPD y LOPDGDD).

5. Modelo Entidad-Relación y Modelo Relacional

5.1 Justificación del Modelo E/R

El modelo entidad-relación de **SpotMap** surge del análisis de los requisitos funcionales del sistema. A continuación se explica el origen de cada entidad y relación del modelo.

5.1.1 Origen de las Entidades

USERS (Usuarios) - **Origen**: Requisito RF1, RF2, RF3 - Sistema de autenticación - **Justificación**: Es necesaria para identificar usuarios, establecer autoría de contenido y gestionar permisos según roles (usuario, moderador, administrador)

SPOTS (Puntos Fotográficos) - **Origen**: Requisito RF4, RF5, RF6, RF7 - Funcionalidad principal - **Justificación**: Entidad central del sistema. Representa localizaciones fotográficas con información geoespacial, descriptiva y multimedia

CATEGORIES (Categorías) - **Origen**: Requisito RF7, RF8 - Clasificación de contenido - **Justificación**: Organiza spots por tipo (mirador, playa, urbano, montaña) facilitando navegación y filtrado

COMMENTS (Comentarios) - **Origen**: Requisito RF10 - Interacción social - **Justificación**: Permite a usuarios compartir experiencias y consejos sobre cada spot. Incluye soporte para respuestas anidadas (hilos de conversación)

RATINGS (Valoraciones) - **Origen**: Requisito RF7 - Sistema de calidad - **Justificación**: Sistema de valoración 1-5 estrellas para medir satisfacción y generar ranking de popularidad

FAVORITES (Favoritos) - **Origen**: Requisito RF9 - Colecciones personales - **Justificación**: Permite a usuarios guardar spots de interés para planificar rutas fotográficas

REPORTS (Reportes) - **Origen**: Requisito RF11, RF12 - Moderación de contenido - **Justificación**: Sistema de control de calidad para gestionar contenido inapropiado o duplicado

TAGS (Etiquetas) - **Origen**: Requisito RF8 - Búsqueda avanzada - **Justificación**: Sistema flexible de palabras clave (hashtags) para mejorar descubrimiento de spots

IMAGES (Imágenes) - **Origen**: Requisito RF4 - Contenido multimedia - **Justificación**: Gestiona múltiples imágenes por spot con metadatos y orden de visualización

NOTIFICATIONS (Notificaciones) - **Origen**: Requisito RF15 - Comunicación con usuarios - **Justificación**: Sistema de alertas para mantener usuarios informados de actividad relevante (comentarios, likes, reportes)

5.1.2 Origen de las Relaciones

USER → SPOT (1:N - "crea") - Un usuario puede crear múltiples spots - Cada spot pertenece a un único autor - Necesaria para establecer autoría y permisos de edición

SPOT → CATEGORY (N:1 - "pertenece a") - Cada spot tiene una categoría principal - Una categoría agrupa múltiples spots - Facilita filtrado eficiente en el mapa

USER → COMMENT (1:N - "escribe") - Un usuario puede escribir múltiples comentarios - Identifica al autor de cada comentario

SPOT → COMMENT (1:N - "tiene") - Un spot puede tener múltiples comentarios - Centraliza conversaciones sobre cada spot

COMMENT → COMMENT (1:N - "responde a") - Auto-relación para respuestas anidadas - Permite conversaciones estructuradas en hilos

USER → RATING (1:N - "valora") - Un usuario puede valorar múltiples spots - Restricción: Un usuario solo puede valorar cada spot una vez

SPOT → RATING (1:N - "recibe") - Un spot puede recibir múltiples valoraciones - Se calcula media automáticamente

USER → FAVORITE (1:N - "guarda") - Un usuario puede tener múltiples favoritos - Implementa colecciones personales

SPOT → FAVORITE (1:N - "es favorito de") - Un spot puede ser favorito de múltiples usuarios - Mide popularidad real

USER → REPORT (1:N - "reporta") - Un usuario puede crear múltiples reportes - Previene spam con validaciones

SPOT → REPORT (1:N - "es reportado") - Un spot puede ser reportado múltiples veces - Prioriza spots con más reportes

COMMENT → REPORT (1:N - "es reportado") - Un comentario puede ser reportado múltiples veces - Sistema de moderación de comentarios

SPOT ↔ TAG (N:M - "está etiquetado con") - Un spot puede tener múltiples tags - Un tag puede estar en múltiples spots - Implementado mediante tabla intermedia SPOT_TAGS

SPOT → IMAGE (1:N - “contiene”) - Un spot puede tener múltiples imágenes - Permite galerías fotográficas

USER → NOTIFICATION (1:N - “recibe”) - Un usuario puede recibir múltiples notificaciones - Sistema de alertas personalizado

5.2 Diagrama Entidad-Relación

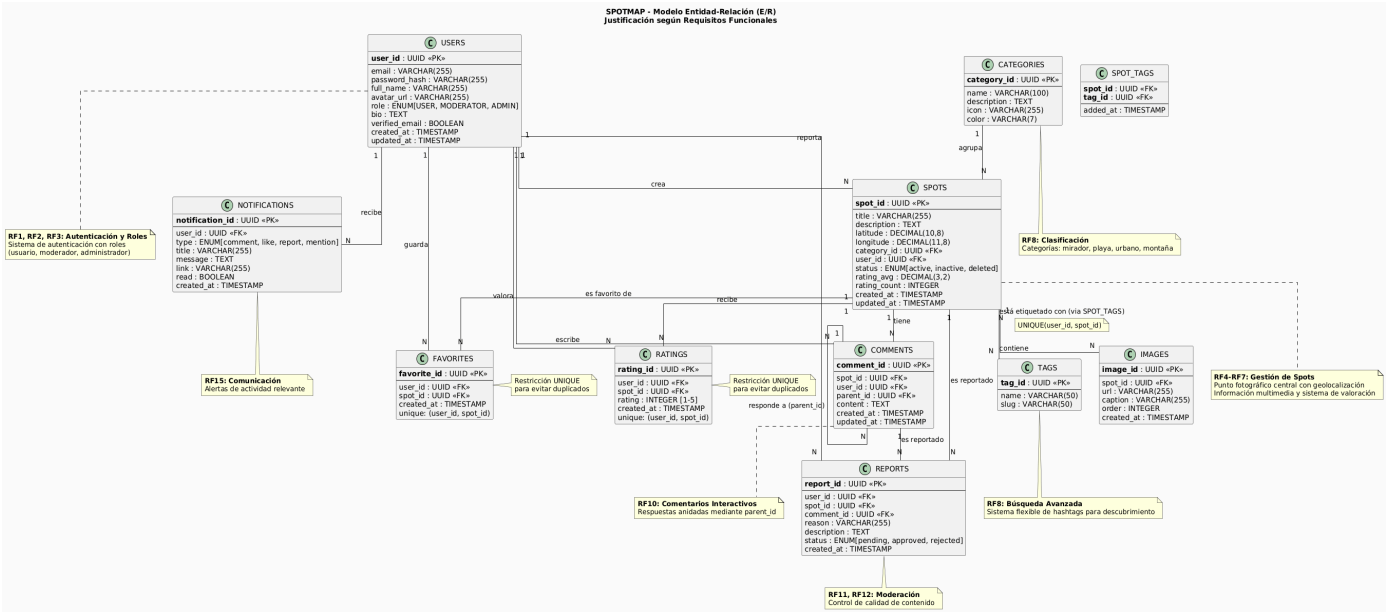


Diagrama E/R de SpotMap

Figura 5.1: Diagrama Entidad-Relación (E/R) completo de SpotMap mostrando 11 entidades, 16 relaciones, cardinalidades (1:N y N:M), y flujos de datos entre USERS, SPOTS, CATEGORIES, COMMENTS, RATINGS, FAVORITES, TAGS, SPOT_TAGS, IMAGES, REPORTS, y NOTIFICATIONS.

5.3 Explicación del Origen de los Atributos

Tabla USERS

Atributo	Tipo de Dato	Origen	Justificación
id	UUID	Identificador único	Clave primaria. Se usa UUID por seguridad (no secuencial) y compatibilidad con Supabase
email	VARCHAR(255) UNIQUE	RF1 - Registro	Identificador único del usuario para login. Debe ser único y válido
password	VARCHAR(255)	RF1 - Registro	Hash de la contraseña (bcrypt/argon2). Nunca se

Atributo	Tipo de Dato	Origen	Justificación
			almacena en texto plano
full_name	VARCHAR(100)	RF1 - Registro	Nombre completo del usuario para personalización de la interfaz
role	ENUM	Gestión de permisos	Valores: ‘user’, ‘moderator’, ‘admin’. Control de acceso basado en roles (RBAC)
avatar_url	VARCHAR(500)	Personalización	URL de la imagen de perfil. Puede ser externa o de Supabase Storage
email_verified	BOOLEAN	Seguridad	Indica si el email ha sido verificado. Previene cuentas falsas
created_at	TIMESTAMP	Auditoría	Fecha de creación de la cuenta. Generado automáticamente
updated_at	TIMESTAMP	Auditoría	Última actualización del perfil. Se actualiza automáticamente

Tabla SPOTS

Atributo	Tipo de Dato	Origen	Justificación
id	UUID	Identificador único	Clave primaria autogenerada
user_id	UUID (FK)	RF4 - Crear spot	Relación con el usuario creador. ON DELETE CASCADE
category_id	INT (FK)	RF7 - Clasificación	Categoría del spot (mirador, playa, etc.). ON DELETE SET NULL
title	VARCHAR(200)	RF4 - Crear spot	Nombre descriptivo del lugar fotográfico. Campo obligatorio
description	TEXT	RF4 - Crear spot	Descripción detallada, consejos, horarios. Campo opcional
latitude	DECIMAL(10,8)	RF4 - Geolocalización	Coordenada norte-sur. Rango: -90 a 90. Precisión ~1cm
longitude	DECIMAL(11,8)	RF4 - Geolocalización	Coordenada este-oeste. Rango: -180 a 180. Precisión ~1cm
image_url	VARCHAR(500)	RF4 - Multimedia	URL de la imagen principal del spot

Atributo	Tipo de Dato	Origen	Justificación
address	VARCHAR(500)	RF7 - Búsqueda	Dirección legible obtenida por geocoding inverso
rating_avg	DECIMAL(2,1)	Calculado	Media de valoraciones (1.0 - 5.0). Campo desnormalizado para rendimiento
rating_count	INT	Contador	Número total de valoraciones. Campo desnormalizado
views	INT	Métricas	Contador de visualizaciones del spot
status	ENUM	Moderación	Valores: 'active', 'pending', 'deleted'. Control del estado del contenido
created_at	TIMESTAMP	Auditoría	Fecha de creación del spot
updated_at	TIMESTAMP	Auditoría	Última modificación del spot

Tabla CATEGORIES

Atributo	Tipo de Dato	Origen	Justificación
id	INT	Identificador único	Clave primaria secuencial. Se usa INT por ser catálogo pequeño
name	VARCHAR(50) UNIQUE	RF7 - Clasificación	Nombre de la categoría. Ej: "Mirador", "Playa", "Urbano"
slug	VARCHAR(50) UNIQUE	SEO	Versión URL-friendly. Ej: "mirador-natural"
icon	VARCHAR(50)	UI/UX	Nombre del icono de Bootstrap Icons. Ej: "bi-mountain"
color	VARCHAR(7)	UI/UX	Color hex para identificación visual. Ej: "#10b981"
description	TEXT	Información	Descripción del tipo de spots que incluye la categoría

Tabla COMMENTS

Atributo	Tipo de Dato	Origen	Justificación
id	UUID	Identificador único	Clave primaria autogenerada
user_id	UUID (FK)	RF10 - Comentar	Usuario autor del comentario. ON DELETE CASCADE

Atributo	Tipo de Dato	Origen	Justificación
spot_id	UUID (FK)	RF10 - Comentar	Spot al que pertenece. ON DELETE CASCADE
parent_id	UUID (FK) nullable	Conversaciones	ID del comentario padre. NULL si es comentario raíz. Auto-relación
text	TEXT	RF10 - Contenido	Texto del comentario. Campo obligatorio
likes	INT	Interacción	Número de likes recibidos. Default: 0
created_at	TIMESTAMP	Auditoría	Fecha de creación del comentario
updated_at	TIMESTAMP	Auditoría	Fecha de última edición

Tabla RATINGS

Atributo	Tipo de Dato	Origen	Justificación
id	INT	Identificador único	Clave primaria secuencial
user_id	UUID (FK)	Valoración	Usuario que valoró. ON DELETE CASCADE
spot_id	UUID (FK)	Valoración	Spot valorado. ON DELETE CASCADE
rating	TINYINT	RF7 - Sistema valoración	Puntuación 1-5 estrellas. CHECK (rating >= 1 AND rating <= 5)
created_at	TIMESTAMP	Auditoría	Fecha de la valoración

Restricción: UNIQUE(user_id, spot_id) - Un usuario solo puede valorar cada spot una vez

Tabla FAVORITES

Atributo	Tipo de Dato	Origen	Justificación
id	INT	Identificador único	Clave primaria secuencial
user_id	UUID (FK)	RF9 - Favoritos	Usuario que guardó el favorito. ON DELETE CASCADE
spot_id	UUID (FK)	RF9 - Favoritos	Spot guardado. ON DELETE CASCADE
created_at	TIMESTAMP	Auditoría	Fecha en que se marcó como

Atributo	Tipo de Dato	Origen	Justificación
			favorito

Restricción: UNIQUE(user_id, spot_id) - Un usuario no puede guardar el mismo spot dos veces

Tabla REPORTS

Atributo	Tipo de Dato	Origen	Justificación
id	UUID	Identificador único	Clave primaria autogenerada
user_id	UUID (FK)	RF12 - Reportar	Usuario que realizó el reporte. ON DELETE CASCADE
spot_id	UUID (FK) nullable	RF12 - Contenido	Spot reportado. NULL si se reporta un comentario
comment_id	UUID (FK) nullable	RF12 - Contenido	Comentario reportado. NULL si se reporta un spot
reason	ENUM	RF12 - Moderación	Valores: ‘spam’, ‘inappropriate’, ‘duplicate’, ‘fake’, ‘other’
description	TEXT	RF12 - Detalles	Descripción detallada del problema
status	ENUM	RF11 - Moderación	Valores: ‘pending’, ‘reviewed’, ‘resolved’, ‘dismissed’
created_at	TIMESTAMP	Auditoría	Fecha del reporte
resolved_at	TIMESTAMP nullable	Auditoría	Fecha de resolución
resolved_by	UUID (FK) nullable	Auditoría	Moderador que resolvió el reporte

Tabla TAGS

Atributo	Tipo de Dato	Origen	Justificación
id	INT	Identificador único	Clave primaria secuencial
name	VARCHAR(50) UNIQUE	RF8 - Búsqueda	Nombre del tag (hashtag sin #). Ej: “atardecer”
slug	VARCHAR(50) UNIQUE	SEO	Versión URL-friendly del nombre
usage_count	INT	Métricas	Contador de cuántos spots tienen este tag

Tabla SPOT_TAGS (Tabla intermedia N:M)

Atributo	Tipo de Dato	Origen	Justificación
id	INT	Identificador único	Clave primaria secuencial
spot_id	UUID (FK)	Relación N:M	Spot etiquetado. ON DELETE CASCADE
tag_id	INT (FK)	Relación N:M	Tag asociado. ON DELETE CASCADE

Restricción: UNIQUE(spot_id, tag_id) - Un spot no puede tener el mismo tag duplicado

Tabla IMAGES

Atributo	Tipo de Dato	Origen	Justificación
id	UUID	Identificador único	Clave primaria autogenerada
spot_id	UUID (FK)	RF4 - Multimedia	Spot al que pertenece. ON DELETE CASCADE
url	VARCHAR(500)	Almacenamiento	URL de la imagen en Supabase Storage o CDN
caption	VARCHAR(200)	Accesibilidad	Descripción alternativa de la imagen
order	INT	UI/UX	Orden de visualización en galería (1, 2, 3…)
created_at	TIMESTAMP	Auditoría	Fecha de subida de la imagen

Tabla NOTIFICATIONS

Atributo	Tipo de Dato	Origen	Justificación
id	UUID	Identificador único	Clave primaria autogenerada
user_id	UUID (FK)	RF15 - Notificaciones	Usuario destinatario. ON DELETE CASCADE
type	ENUM	RF15 - Tipos	Valores: ‘comment’, ‘like’, ‘favorite’, ‘report’, ‘system’
title	VARCHAR(100)	UI/UX	Título corto de la notificación
message	TEXT	RF15 - Contenido	Mensaje completo de la notificación
link	VARCHAR(500)	Navegación	URL de destino al hacer click

Atributo	Tipo de Dato	Origen	Justificación
read	BOOLEAN	Estado	Indica si la notificación ha sido leída. Default: false
created_at	TIMESTAMP	Auditoría	Fecha de creación de la notificación

5.4 Elección de Claves Primarias

Para el diseño de SpotMap se ha optado por una **estrategia híbrida** que combina UUID e INT según las necesidades de cada tabla.

UUID (Universally Unique Identifier)

Se utiliza en: - users.id - spots.id - comments.id - images.id - notifications.id - reports.id

Justificación:

1. **Escalabilidad distribuida**: Los UUID se pueden generar en el cliente sin necesidad de consultar la base de datos, evitando colisiones incluso en sistemas distribuidos
2. **Seguridad**: Al no ser secuenciales, no exponen información sobre el número total de registros en la tabla (no se puede saber cuántos usuarios hay contando IDs)
3. **Compatibilidad con Supabase**: Supabase utiliza UUID como tipo de dato por defecto para claves primarias en su sistema de autenticación
4. **Sincronización offline**: Facilita la implementación de funcionalidades offline ya que los IDs se pueden generar localmente sin conflictos
5. **No predecibilidad**: Los IDs no son adivinables, lo que aumenta la seguridad (un usuario no puede “adivinar” el ID de otro usuario o spot)

INT Autoincremental

Se utiliza en: - categories.id - tags.id - ratings.id - favorites.id - spot_tags.id

Justificación:

1. **Catálogos pequeños**: Categorías y tags son conjuntos limitados y relativamente estáticos
2. **Eficiencia de indexación**: Los enteros son más rápidos en operaciones de JOIN que los UUID, especialmente con grandes volúmenes de datos
3. **Menor consumo de espacio**: INT(11) ocupa 4 bytes mientras que UUID ocupa 16 bytes (reducción del 75% de espacio)
4. **Legibilidad**: Es más fácil referenciar “categoría 3” que un UUID largo en logs y debugging
5. **Orden natural**: El autoincremento proporciona un orden cronológico natural útil para ordenar por antigüedad

Aspecto	UUID	INT Autoincremental
Tamaño	16 bytes	4 bytes

Aspecto	UUID	INT Autoincremental
Generación	Cliente/Servidor	Solo servidor
Orden	Aleatorio	Secuencial
Seguridad	Alta (no predecible)	Baja (predecible)
Rendimiento JOIN	Menor	Mayor
Uso recomendado	Entidades principales	Catálogos y relaciones

5.5 Modelo Relacional

5.5.1 Paso del Modelo E/R al Modelo Relacional

El proceso de transformación del modelo Entidad-Relación al Modelo Relacional sigue las siguientes reglas:

Regla 1: Transformación de Entidades

Cada entidad del modelo E/R se convierte en una tabla con las siguientes características: - El nombre de la entidad se convierte en el nombre de la tabla - Cada atributo de la entidad se convierte en una columna - La clave primaria de la entidad se convierte en PRIMARY KEY de la tabla - Los atributos obligatorios se marcan como NOT NULL

Ejemplo: La entidad USERS se transforma en:

```
CREATE TABLE users (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    email VARCHAR(255) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    full_name VARCHAR(100) NOT NULL,
    role VARCHAR(20) DEFAULT 'user',
    avatar_url VARCHAR(500),
    email_verified BOOLEAN DEFAULT false,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Regla 2: Transformación de Relaciones 1:N

Las relaciones uno a muchos (1:N) se implementan mediante claves foráneas: - Se añade una columna en la tabla del lado “N” que referencia la PRIMARY KEY de la tabla del lado “1” - Esta columna se marca como FOREIGN KEY - Se definen las acciones ON DELETE y ON UPDATE según las reglas de negocio

Ejemplo: La relación USER → SPOT (1:N) se transforma en:

```
CREATE TABLE spots (
    id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
    user_id UUID NOT NULL,
    -- otros atributos...
```

```
  CONSTRAINT fk_spot_user
    FOREIGN KEY (user_id)
    REFERENCES users(id)
    ON DELETE CASCADE
);
```

Significado de ON DELETE CASCADE: Cuando se elimina un usuario, todos sus spots se eliminan automáticamente (borrado en cascada).

Regla 3: Transformación de Relaciones N:M

Las relaciones muchos a muchos (N:M) requieren una tabla intermedia: - Se crea una nueva tabla con el nombre de ambas entidades (ej: spot_tags) - Esta tabla contiene dos claves foráneas que referencian las tablas relacionadas - La clave primaria puede ser compuesta (ambas FKs) o un ID independiente - Se añade constraint UNIQUE para evitar duplicados

Ejemplo: La relación SPOT ↔ TAG (N:M) se transforma en:

```
CREATE TABLE spot_tags (
  id INT PRIMARY KEY AUTO_INCREMENT,
  spot_id UUID NOT NULL,
  tag_id INT NOT NULL,

  CONSTRAINT fk_spot_tags_spot
    FOREIGN KEY (spot_id)
    REFERENCES spots(id)
    ON DELETE CASCADE,

  CONSTRAINT fk_spot_tags_tag
    FOREIGN KEY (tag_id)
    REFERENCES tags(id)
    ON DELETE CASCADE,

  CONSTRAINT unique_spot_tag
    UNIQUE (spot_id, tag_id)
);
```

Regla 4: Auto-relaciones

Las auto-relaciones (una entidad relacionada consigo misma) se implementan con FK a la misma tabla:

Ejemplo: COMMENT → COMMENT (respuestas anidadas):

```
CREATE TABLE comments (
  id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  user_id UUID NOT NULL,
  spot_id UUID NOT NULL,
  parent_id UUID NULL,  -- Auto-relación
  text TEXT NOT NULL,
```

```
    CONSTRAINT fk_comment_parent
        FOREIGN KEY (parent_id)
        REFERENCES comments(id)
        ON DELETE CASCADE
);
```

Explicación: parent_id puede ser NULL (comentario raíz) o referenciar a otro comentario de la misma tabla (respuesta).

5.5.2 Esquema Relacional Completo

Tabla: USERS

```
users (
    id: UUID [PK],
    email: VARCHAR(255) [UNIQUE, NOT NULL],
    password: VARCHAR(255) [NOT NULL],
    full_name: VARCHAR(100) [NOT NULL],
    role: VARCHAR(20) [DEFAULT 'user'],
    avatar_url: VARCHAR(500),
    email_verified: BOOLEAN [DEFAULT false],
    created_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP],
    updated_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP]
)
```

Tabla: CATEGORIES

```
categories (
    id: INT [PK, AUTO_INCREMENT],
    name: VARCHAR(50) [UNIQUE, NOT NULL],
    slug: VARCHAR(50) [UNIQUE, NOT NULL],
    icon: VARCHAR(50),
    color: VARCHAR(7),
    description: TEXT
)
```

Tabla: SPOTS

```
spots (
    id: UUID [PK],
    user_id: UUID [FK → users.id, NOT NULL, ON DELETE CASCADE],
    category_id: INT [FK → categories.id, ON DELETE SET NULL],
    title: VARCHAR(200) [NOT NULL],
    description: TEXT,
    latitude: DECIMAL(10,8) [NOT NULL, CHECK >= -90 AND <= 90],
    longitude: DECIMAL(11,8) [NOT NULL, CHECK >= -180 AND <= 180],
    image_url: VARCHAR(500),
    address: VARCHAR(500),
    rating_avg: DECIMAL(2,1) [DEFAULT 0, CHECK >= 0 AND <= 5],
    rating_count: INT [DEFAULT 0],
    views: INT [DEFAULT 0],
```

```
    status: VARCHAR(20) [DEFAULT 'active'],
    created_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP],
    updated_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP]
)
```

Tabla: COMMENTS

```
comments (
    id: UUID [PK],
    user_id: UUID [FK → users.id, NOT NULL, ON DELETE CASCADE],
    spot_id: UUID [FK → spots.id, NOT NULL, ON DELETE CASCADE],
    parent_id: UUID [FK → comments.id, ON DELETE CASCADE],
    text: TEXT [NOT NULL],
    likes: INT [DEFAULT 0],
    created_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP],
    updated_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP]
)
```

Tabla: RATINGS

```
ratings (
    id: INT [PK, AUTO_INCREMENT],
    user_id: UUID [FK → users.id, NOT NULL, ON DELETE CASCADE],
    spot_id: UUID [FK → spots.id, NOT NULL, ON DELETE CASCADE],
    rating: TINYINT [NOT NULL, CHECK >= 1 AND <= 5],
    created_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP],

    UNIQUE (user_id, spot_id)
)
```

Tabla: FAVORITES

```
favorites (
    id: INT [PK, AUTO_INCREMENT],
    user_id: UUID [FK → users.id, NOT NULL, ON DELETE CASCADE],
    spot_id: UUID [FK → spots.id, NOT NULL, ON DELETE CASCADE],
    created_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP],

    UNIQUE (user_id, spot_id)
)
```

Tabla: REPORTS

```
reports (
    id: UUID [PK],
    user_id: UUID [FK → users.id, NOT NULL, ON DELETE CASCADE],
    spot_id: UUID [FK → spots.id, ON DELETE CASCADE],
    comment_id: UUID [FK → comments.id, ON DELETE CASCADE],
    reason: VARCHAR(20) [NOT NULL],
    description: TEXT,
    status: VARCHAR(20) [DEFAULT 'pending'],
```

```
    created_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP],
    resolved_at: TIMESTAMP,
    resolved_by: UUID [FK → users.id, ON DELETE SET NULL]
)

Tabla: TAGS

tags (
    id: INT [PK, AUTO_INCREMENT],
    name: VARCHAR(50) [UNIQUE, NOT NULL],
    slug: VARCHAR(50) [UNIQUE, NOT NULL],
    usage_count: INT [DEFAULT 0]
)

Tabla: SPOT_TAGS (tabla intermedia)

spot_tags (
    id: INT [PK, AUTO_INCREMENT],
    spot_id: UUID [FK → spots.id, NOT NULL, ON DELETE CASCADE],
    tag_id: INT [FK → tags.id, NOT NULL, ON DELETE CASCADE],

    UNIQUE (spot_id, tag_id)
)

Tabla: IMAGES

images (
    id: UUID [PK],
    spot_id: UUID [FK → spots.id, NOT NULL, ON DELETE CASCADE],
    url: VARCHAR(500) [NOT NULL],
    caption: VARCHAR(200),
    order: INT [DEFAULT 1],
    created_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP]
)

Tabla: NOTIFICATIONS

notifications (
    id: UUID [PK],
    user_id: UUID [FK → users.id, NOT NULL, ON DELETE CASCADE],
    type: VARCHAR(20) [NOT NULL],
    title: VARCHAR(100) [NOT NULL],
    message: TEXT,
    link: VARCHAR(500),
    read: BOOLEAN [DEFAULT false],
    created_at: TIMESTAMP [DEFAULT CURRENT_TIMESTAMP]
)
```

5.5.3 Diagrama Relacional

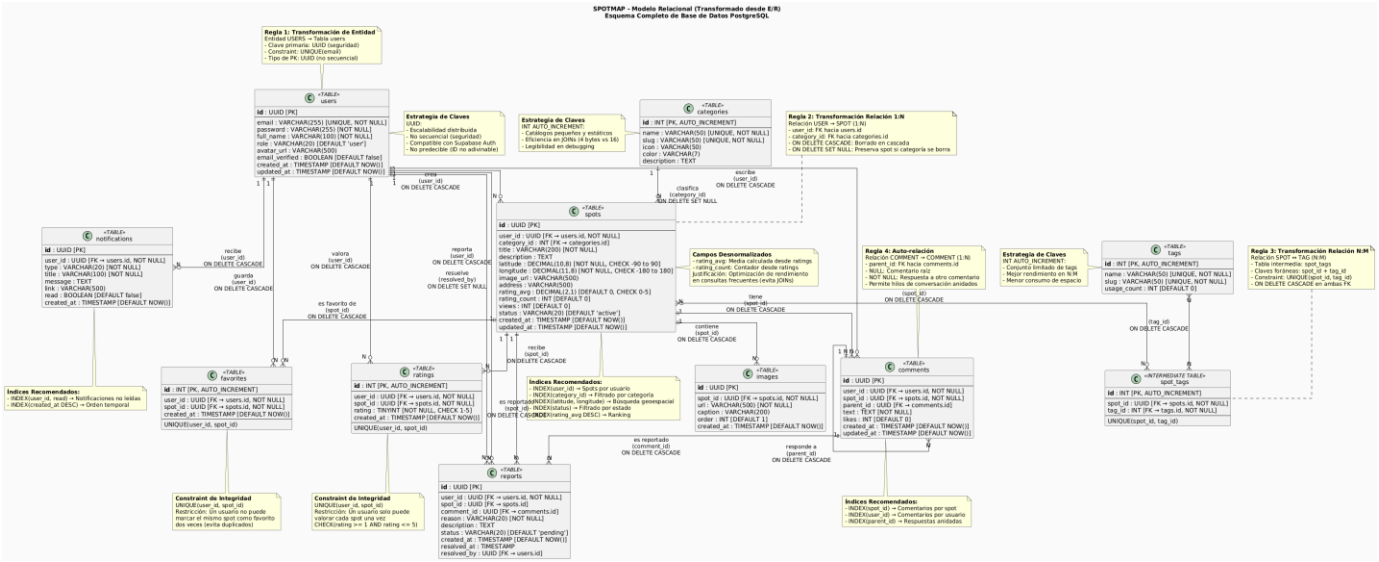


Diagrama Relacional de SpotMap

Figura 5.2: Esquema Relacional detallado con 11 tablas PostgreSQL, incluyendo tipos de datos, restricciones (NOT NULL, UNIQUE, FK), índices, y relaciones ON DELETE CASCADE/SET NULL. Base de datos normalizada a 3NF.

5.5.4 Explicación de ON DELETE y ON UPDATE

Las cláusulas ON DELETE y ON UPDATE definen el comportamiento automático cuando se elimina o actualiza un registro referenciado:

ON DELETE CASCADE

Uso: ON DELETE CASCADE

Significado: Cuando se elimina un registro padre, se eliminan automáticamente todos los registros hijos relacionados (borrado en cascada).

Ejemplos en SpotMap:

-- Si se elimina un usuario, se eliminan todos sus spots
spots.user_id → users.**id ON DELETE CASCADE**

-- Si se elimina un spot, se eliminan todos sus comentarios
comments.spot_id → spots.**id ON DELETE CASCADE**

-- Si se elimina un spot, se eliminan todas sus valoraciones
ratings.spot_id → spots.**id ON DELETE CASCADE**

Justificación: Mantiene la integridad referencial automáticamente. Si un usuario se da de baja, todo su contenido debe eliminarse.

ON DELETE SET NULL

Uso: ON DELETE SET NULL

Significado: Cuando se elimina un registro padre, las claves foráneas de los registros hijos se establecen a NULL (en lugar de eliminarlos).

Ejemplo en SpotMap:

-- Si se elimina una categoría, los spots mantienen pero sin categoría
spots.category_id → categories.**id ON DELETE SET NULL**

Justificación: Se prefiere mantener los spots aunque su categoría desaparezca. La categoría puede asignarse posteriormente.

Relación	ON DELETE	Justificación
spots.user_id → users	CASCADE	El contenido de un usuario se elimina con él
spots.category_id → categories	SET NULL	Los spots se mantienen sin categoría
comments.user_id → users	CASCADE	Los comentarios se eliminan con el usuario
comments.spot_id → spots	CASCADE	Los comentarios se eliminan con el spot
comments.parent_id → comments	CASCADE	Las respuestas se eliminan con el comentario padre
ratings.user_id → users	CASCADE	Las valoraciones se eliminan con el usuario
ratings.spot_id → spots	CASCADE	Las valoraciones se eliminan con el spot
favorites.user_id → users	CASCADE	Los favoritos se eliminan con el usuario
favorites.spot_id → spots	CASCADE	Los favoritos se eliminan con el spot
spot_tags.spot_id → spots	CASCADE	Las etiquetas se eliminan con el spot
images.spot_id → spots	CASCADE	Las imágenes se eliminan con el spot
notifications.user_id → users	CASCADE	Las notificaciones se eliminan con el usuario
reports.resolved_by → users	SET NULL	Se mantiene el reporte aunque el moderador se elimine

5.5.5 Normalización del Modelo

El modelo relacional de SpotMap se encuentra en **Tercera Forma Normal (3NF)**, cumpliendo todos los requisitos de calidad de diseño de bases de datos relacionales.

Primera Forma Normal (1FN)

Requisito: Todos los atributos deben contener valores atómicos (indivisibles).

Cumplimiento en SpotMap: Ninguna columna contiene valores múltiples o listas; cada celda contiene un único valor; no existen grupos repetitivos.

Solución aplicada: La relación N:M entre spots y tags usa tabla intermedia spot_tags, donde cada fila contiene una única asociación spot-tag.

Segunda Forma Normal (2FN)

Requisito: Estar en 1FN + Todos los atributos no clave deben depender completamente de la clave primaria (no dependencias parciales).

Cumplimiento en SpotMap: Todas las tablas tienen claves primarias simples (UUID o INT), no compuestas; no existen dependencias parciales.

Tercera Forma Normal (3NF)

Requisito: Estar en 2FN + No deben existir dependencias transitivas.

Cumplimiento en SpotMap: Todos los atributos no clave dependen únicamente de la clave primaria; no existen dependencias transitivas.

Desnormalización Controlada

Campos desnormalizados intencionalmente: - spots.rating_avg - Media calculada de ratings - spots.rating_count - Contador de ratings - tags.usage_count - Contador de spots con este tag - comments.likes - Contador de likes

Justificación: Estos campos calculados se almacenan por razones de **rendimiento**: evitar JOIN costosos, optimizar consultas frecuentes, mantenimiento automático mediante triggers.

6. Diagramas de Procesos

6.1 Diagramas de Clase

Resumen rápido y visual del modelo de clases.

Clases principales (qué hace cada una): - **User**: usuarios, roles y perfil (admin, moderator, user). - **Spot**: puntos fotográficos, geolocalización, imagen principal, valoraciones. - **Comment**: comentarios e hilos de respuestas anidadas. - **Rating**: valoraciones 1–5 estrellas. - **Category**: clasificación por tipo (mirador, playa, urbano, montaña). - **Favorite**: guardados/colecciones del usuario. - **Otras de soporte**: Tag, Report, Image, Notification.

Relaciones clave: - User 1:N Spot; Spot N:1 Category - Spot 1:N Comment (y Comment 1:N Comment para respuestas) - Spot 1:N Rating; User 1:N Rating - User 1:N Favorite; Favorite N:1 Spot - Spot N:M Tag (vía SPOT_TAGS)

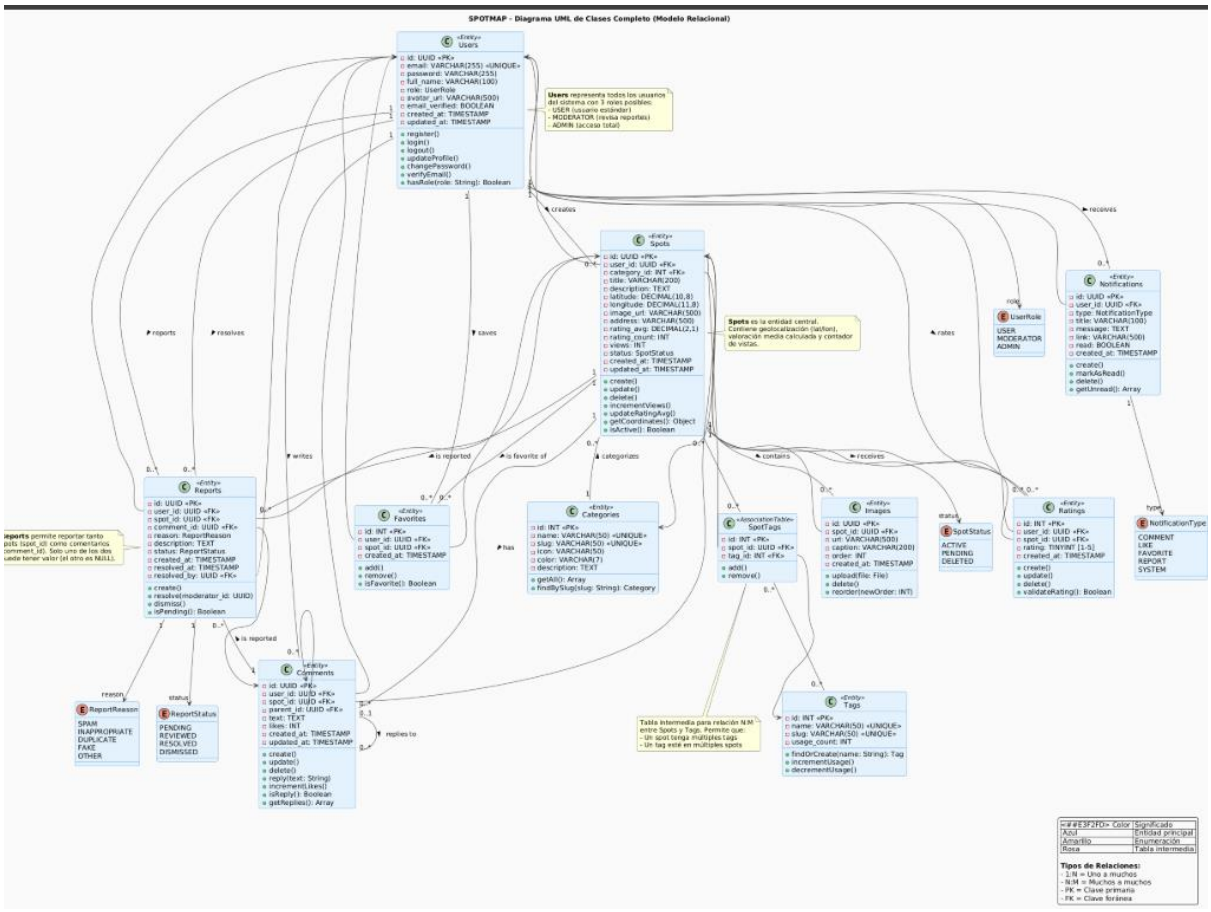


Diagrama UML de Clases de SpotMap

Figura 6.1: Diagrama UML de Clases con 10 clases principales, sus atributos, métodos públicos (+) y privados (-), y relaciones (1:N, N:M, herencia, composición). Incluye todas las clases de dominio y su interacción.

6.2 Casos de Uso

Los casos de uso definen las interacciones entre actores y el sistema.

Actores identificados: - **Visitante**: sin registrar; solo navega y consulta. - **Usuario registrado**: crea, edita, comenta, valora, guarda favoritos. - **Moderador**: revisa reportes, aprueba/oculta contenido. - **Administrador**: gestiona usuarios, roles, estadísticas, configuración global.

Casos de uso principales:

Actor	Caso de Uso	Descripción
Visitante	Navegar mapa	Ver spots en mapa interactivo.
Visitante	Consultar spot	Ver detalles, descripción, comentarios e imágenes de un spot.
Usuario	Registrarse	Crear cuenta con email/contraseña.
Usuario	Iniciar sesión	Acceder con credenciales o OAuth.

Actor	Caso de Uso	Descripción
Usuario	Crear spot	Añadir título, descripción, ubicación, categoría e imagen(s).
Usuario	Editar spot	Modificar datos de spots propios.
Usuario	Eliminar spot	Borrar spots propios.
Usuario	Buscar spots	Búsqueda por texto, categoría, localidad o tags.
Usuario	Valorar spot	Dar puntuación 1-5 estrellas.
Usuario	Comentar	Añadir comentarios y respuestas en hilos.
Usuario	Marcar favorito	Guardar/quitar spots de favoritos.
Usuario	Reportar	Denunciar spot/comentario inapropiado.
Usuario	Ver perfil	Consultar datos personales y spots creados.
Moderador	Revisar reportes	Ver lista de reportes pendientes.
Moderador	Validar contenido	Aprobar, ocultar o eliminar reportados.
Administrador	Gestionar usuarios	Ver, suspender o restablecer usuarios.
Administrador	Ver estadísticas	Panel de métricas (usuarios, spots, actividad).

Diagrama de Casos de Uso de SpotMap

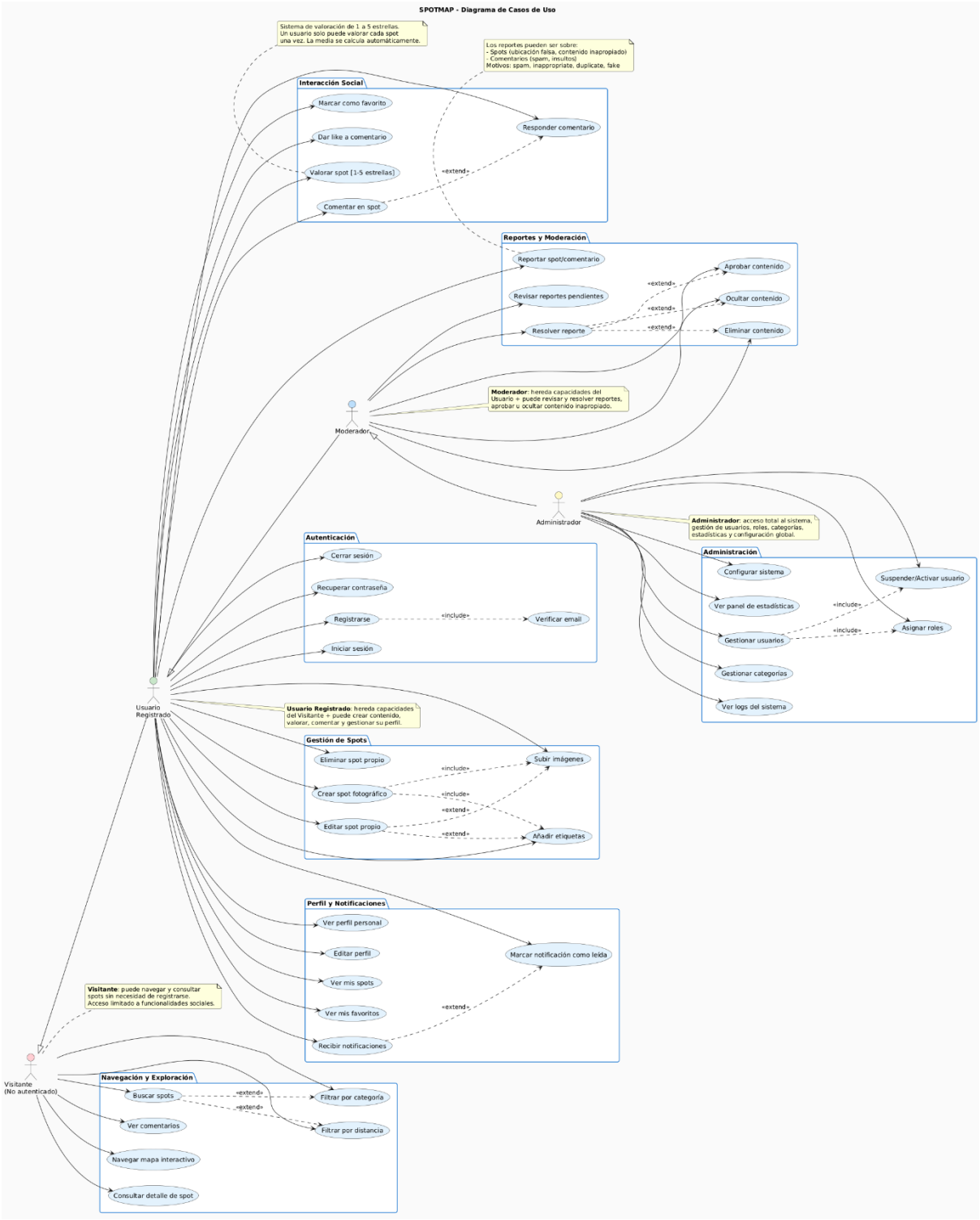


Figura 6.2: Diagrama de Casos de Uso (UML) con 4 actores (Visitante, Usuario Registrado, Moderador, Administrador) y 19 casos de uso, mostrando relaciones de inclusión, extensión y asociación. Cubre flujos de autenticación, exploración, creación, moderación y administración.

6.3 Diagramas de Secuencia

Los diagramas de secuencia muestran la interacción temporal entre los diferentes componentes del sistema (Frontend, API Backend, Base de Datos, Servicios externos) para casos de uso específicos, detallando el flujo completo de datos y las validaciones aplicadas en cada paso.

6.3.1 Registro de Usuario

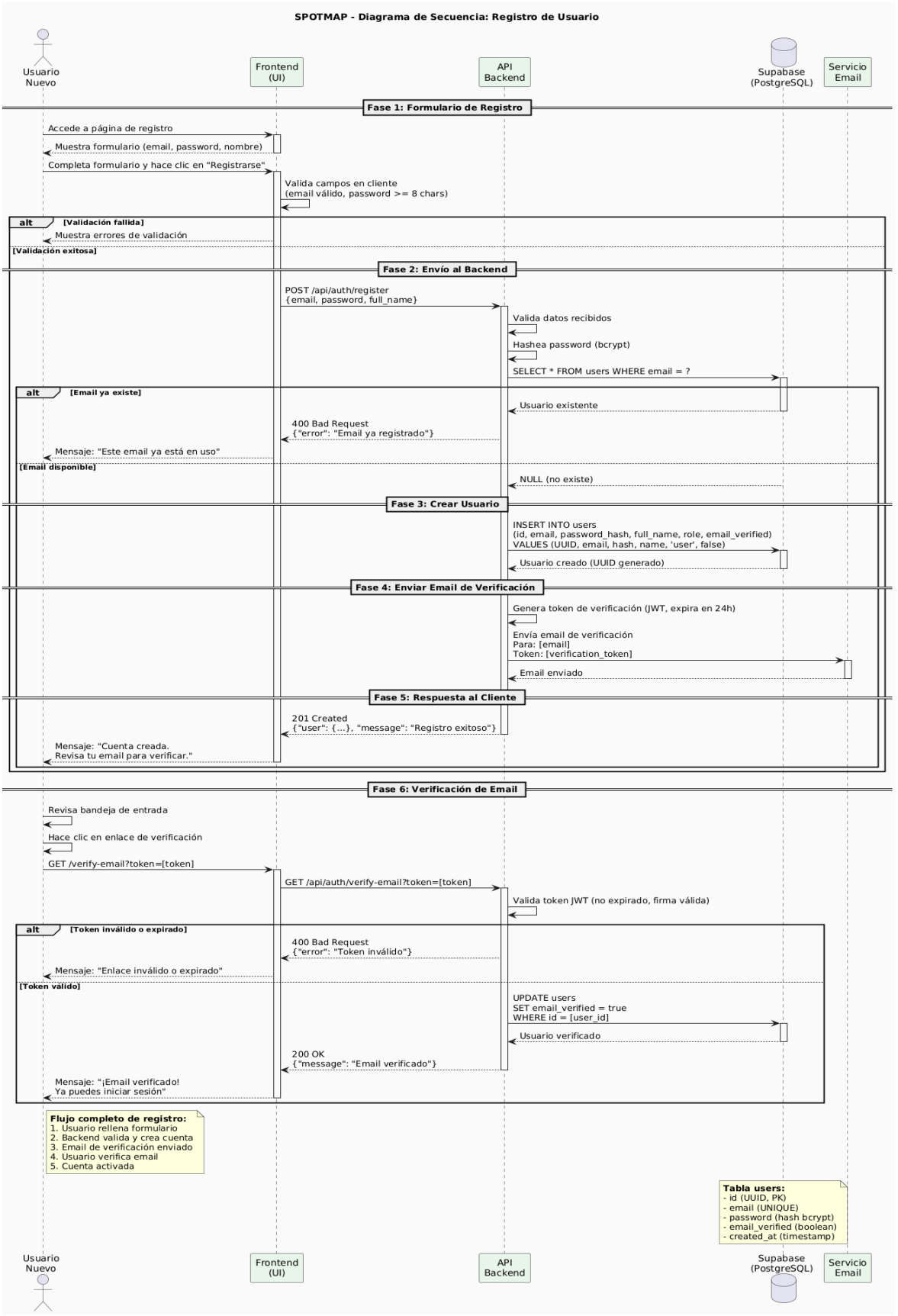


Figura 6.3: Flujo completo de registro de usuario, incluyendo validación de campos en cliente, verificación de email duplicado en base de datos, hash de contraseña con bcrypt, creación de cuenta con UUID, generación de token JWT de verificación (válido 24h), envío de email mediante servicio externo, y activación de cuenta mediante enlace de verificación.

Fases del proceso:

1. Formulario de registro: Usuario completa email, contraseña y nombre completo
2. Validación cliente: Frontend valida formato de email y fortaleza de contraseña
3. Envío al backend: POST /api/auth/register con datos del usuario
4. Validación backend: Verifica email único, hashea contraseña (bcrypt)
5. Creación en BD: INSERT INTO users con email_verified = false
6. Email de verificación: Genera token JWT y envía email con enlace
7. Verificación: Usuario hace clic en enlace, backend valida token y activa cuenta

6.3.2 Creación de Spot Fotográfico

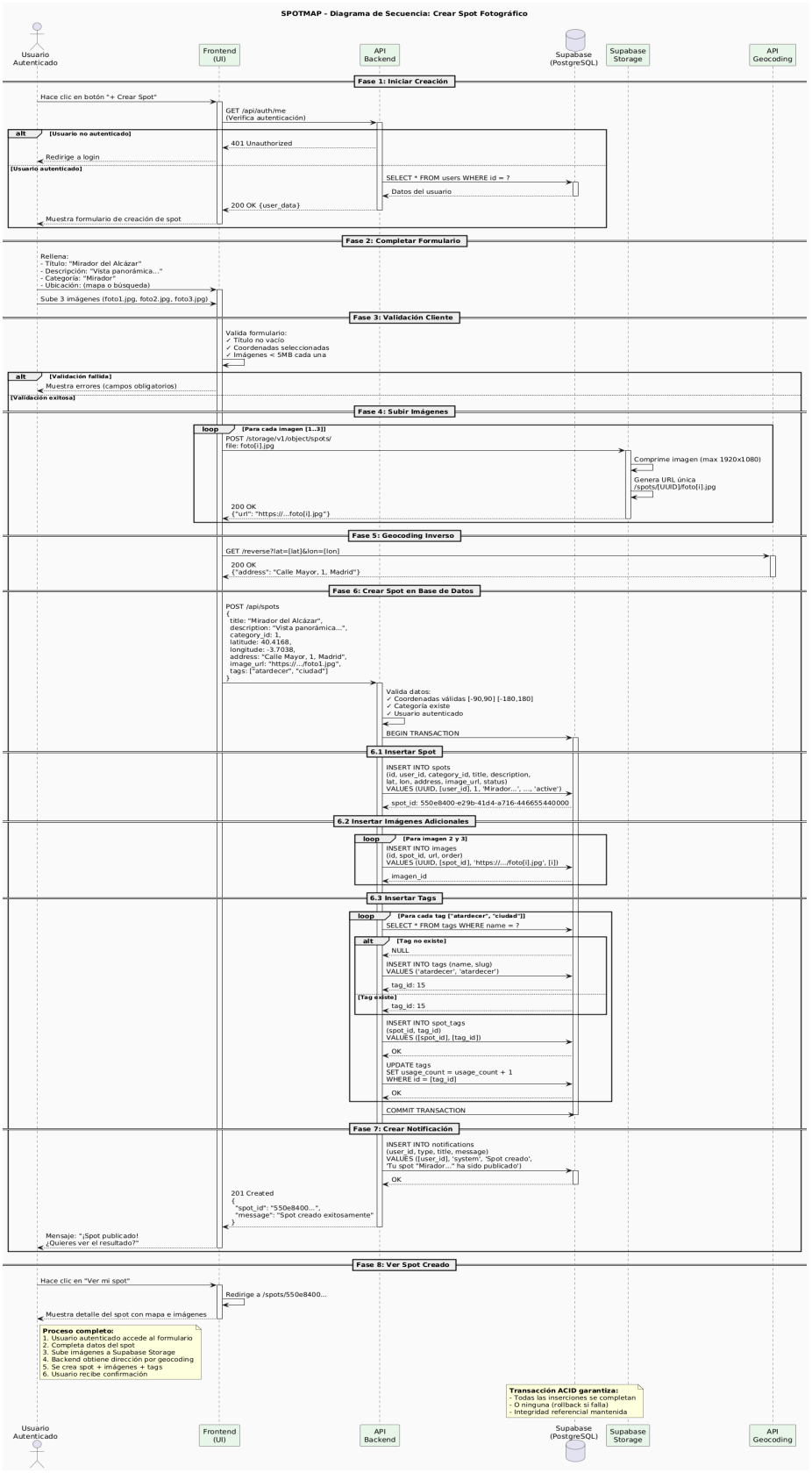


Figura 6.4: Proceso completo de creación de spot, incluyendo verificación de autenticación, validación de formulario multi-paso, subida de imágenes a Supabase Storage con compresión automática (max 1920x1080), geocoding inverso mediante API externa para obtener dirección legible, y transacción ACID en PostgreSQL para garantizar integridad de datos (spot + imágenes + tags).

6.3.3 Valoración de Spot

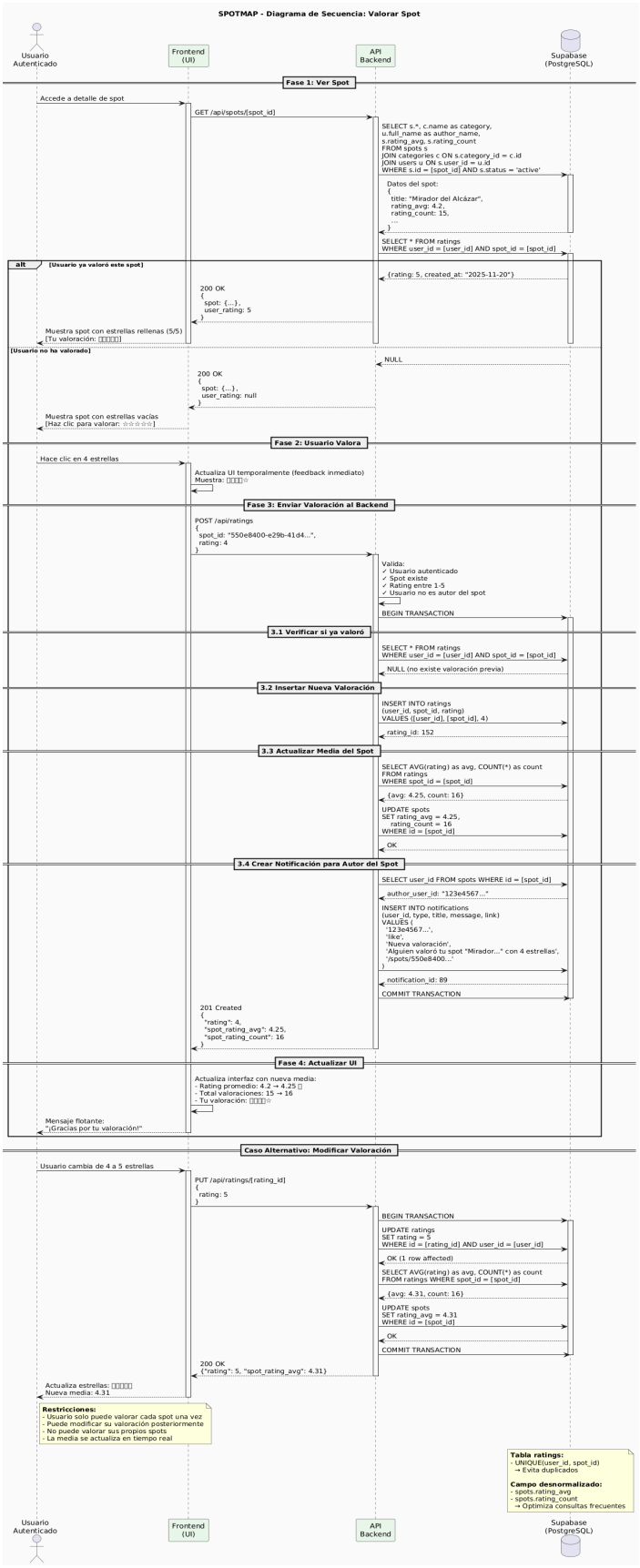


Figura 6.5: Sistema de valoración 1-5 estrellas con actualización en tiempo real de la media ponderada, prevención de duplicados mediante constraint UNIQUE(user_id, spot_id), recálculo automático de rating_avg y rating_count en tabla spots (campos desnormalizados para optimización), notificación al autor del spot, y feedback inmediato en interfaz de usuario.

7. Diseño de Interfaz

7.1 Diagramación y Prototipado

El diseño visual de SpotMap sigue un enfoque **mobile-first**, responsivo y centrado en la experiencia del usuario. La interfaz debe ser intuitiva para usuarios de todos los niveles técnicos.

Pantallas Principales

1. Pantalla de Inicio / Mapa Principal

Propósito: Visualizar spots geolocalizados en un mapa interactivo.

Elementos principales: - Mapa Leaflet/OpenStreetMap ocupando el 80% de la pantalla - Marcadores de colores según categoría (mirador: rojo, playa: azul, urbano: naranja, etc.) - Barra de búsqueda/filtros en la parte superior - Botón flotante (+) para crear nuevo spot - Barra lateral con lista de spots filtrados - Panel de usuario en la esquina superior derecha

Funcionalidades: - Zoom y paneo del mapa - Click en marcador → detalles del spot - Filtros por categoría, distancia, popularidad - Búsqueda en tiempo real

2. Detalle del Spot

Propósito: Mostrar información completa de un spot.

Elementos principales: - Galería de imágenes (carrousel) - Título, descripción y ubicación - Puntuación media (estrellas) - Número de comentarios y valoraciones - Botón “Añadir a favoritos” - Botón “Valorar” - Sección de comentarios con hilos de respuesta - Botón “Reportar” si es inapropiado

Layout: - Imagen principal grande - Información general en tarjeta - Comentarios en sección desplazable - Pie con acciones principales

3. Creación/Edición de Spot

Propósito: Permitir al usuario crear o editar un spot fotográfico.

Formulario con campos: - Título (obligatorio) - Descripción (opcional) - Categoría (selector) - Ubicación (mapa interactivo + búsqueda de dirección) - Imágenes (carga múltiple con arrastrar y soltar) - Etiquetas (input con sugerencias) - Botones: Guardar, Cancelar, Vista previa

Validaciones: - Campos obligatorios marcados - Vista previa en tiempo real - Compresión automática de imágenes - Aviso de cambios sin guardar

4. Perfil de Usuario

Propósito: Mostrar datos personales y contenido del usuario.

Secciones: - Información personal (avatar, nombre, email, fecha registro) - Estadísticas (spots creados, valoraciones, comentarios) - Mis spots (lista con opciones editar/eliminar) - Mis favoritos (spots guardados) - Configuración (contraseña, privacidad, notificaciones) - Opción cerrar sesión

5. Panel de Moderación

Propósito: Permitir a moderadores revisar contenido reportado.

Elementos: - Lista de reportes pendientes con filtros - Detalles del reporte (motivo, fecha, usuario que reporta) - Vista previa del contenido reportado - Acciones: Aprobar, Ocultar, Eliminar, Cerrar reporte - Historial de decisiones

6. Panel de Administración

Propósito: Acceso a gestión global, estadísticas y configuración.

Secciones: - Dashboard con métricas (usuarios, spots, comentarios, reportes) - Gestión de usuarios (listar, suspender, restablecer) - Gestión de categorías (crear, editar, eliminar) - Estadísticas gráficas (Chart.js: usuarios activos, spots por categoría, actividad temporal) - Logs de actividad - Configuración global

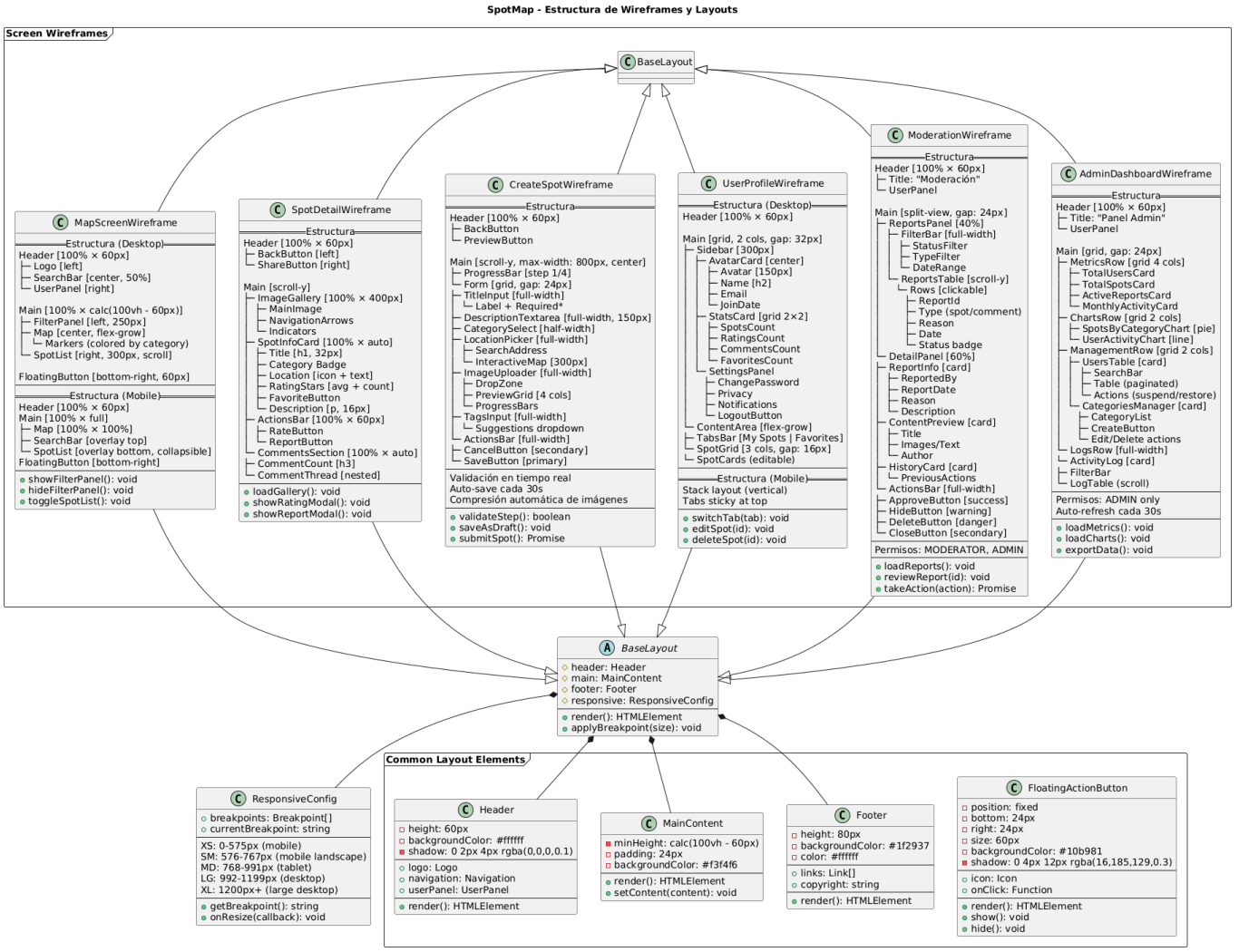


Figura 7.1: Wireframes de 6 pantallas principales: Mapa Interactivo (búsqueda/filtros), Detalle Spot (galería/comentarios), Crear Spot (formulario multi-paso), Perfil Usuario (estadísticas), Panel Moderación (revisión de reportes), y Admin Dashboard (métricas/logs).

7.2 Guías de Estilo

Paleta de Colores

Elemento	Color	Código Hex	Uso
Primario	Verde esmeralda	#10b981	Botones principales, acentos
Secundario	Azul cielo	#3b82f6	Links, botones secundarios
Acento	Naranja cálido	#f97316	Alertas, destacados
Fondo	Blanco	#ffffff	Fondo principal
Fondo secundario	Gris claro	#f3f4f6	Tarjetas, secciones alternas
Texto principal	Gris oscuro	#1f2937	Texto body
Texto secundario	Gris medio	#6b7280	Etiquetas, descripciones
Error	Rojo	#ef4444	Mensajes de error
Éxito	Verde	#22c55e	Mensajes de éxito
Advertencia	Amarillo	#eab308	Mensajes de advertencia

Tipografía

- **Fuente Principal**: Inter, Roboto o -apple-system (system fonts)
- **Tamaños**:
    - H1 (títulos principales): 32px, bold
    - H2 (subtítulos): 24px, bold
    - H3 (secciones): 18px, semi-bold
    - Body (texto normal): 16px, regular
    - Small (labels, ayuda): 14px, regular
    - Tiny (metadata): 12px, regular

Espaciado

- **Padding estándar**: 8px, 16px, 24px, 32px
- **Margen entre secciones**: 24px, 32px
- **Gap entre elementos**: 8px, 12px, 16px

Componentes Reutilizables

Botones

- **Primario**: Fondo verde, texto blanco, 12px padding horizontal, border-radius 6px
- **Secundario**: Borde gris, texto gris, fondo transparente
- **Peligro**: Fondo rojo, texto blanco
- **Estado deshabilitado**: Opacidad 50%, sin hover

Tarjetas

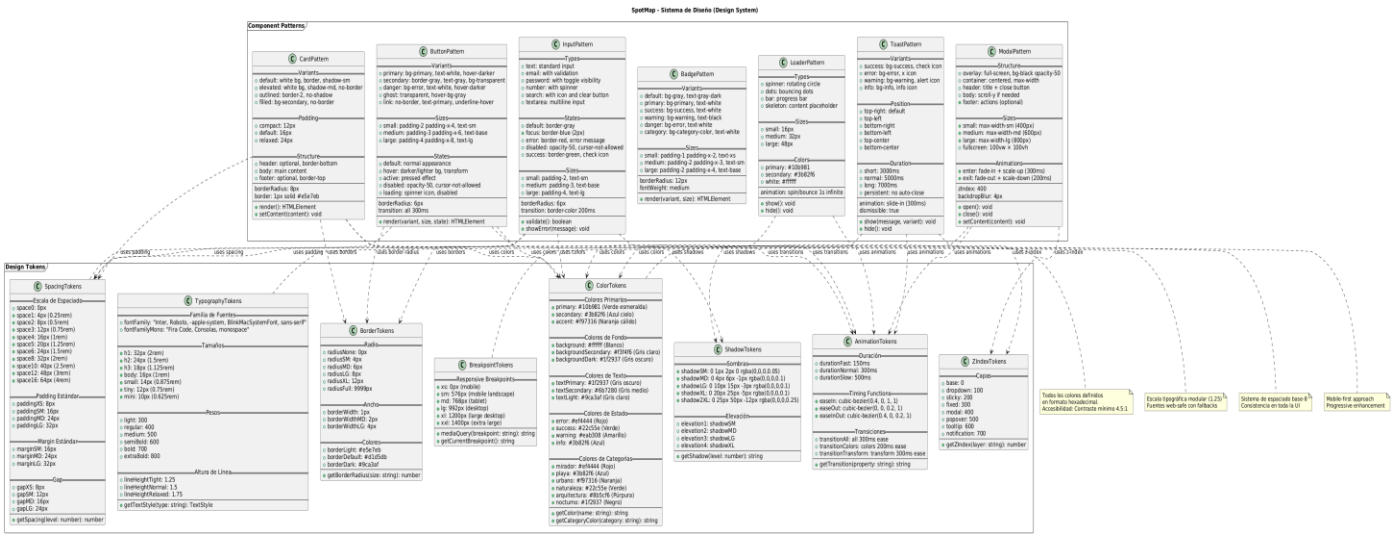
- Fondo blanco, borde gris claro (1px), border-radius 8px, shadow suave
- Padding 16px

Inputs

- Borde gris (1px), focus azul (2px), border-radius 6px
- Padding 12px
- Placeholder en gris claro

Badges

- Fondo de color según categoría, texto blanco, border-radius 12px
- Padding 4px 8px



Guía visual de estilos y componentes de SpotMap

Figura 7.2: Guía completa de estilos (Design System) incluyendo: paleta de colores (#10B981 verde primario, #3B82F6 azul, #F59E0B naranja), tipografía (Segoe UI, escalas 10px-28px), componentes (botones, cards, inputs), espaciado (4-32px), breakpoints responsivos (XS-XL), y efectos de sombra/elevación.

7.3 Mapa de Navegación

Estructura de Sitio (Visitante / Usuario No Registrado)

Inicio (Mapa)
- Explorar spots
  - Filtrar por categoría
  - Buscar por texto
  - Ver detalle del spot
    - Ver comentarios
    - Reportar (requiere login)
- Login / Registro
  - Registro
  - Login
  - Recuperar contraseña
- Información
  - Sobre SpotMap
  - Términos y condiciones
  - Política de privacidad

Estructura de Sitio (Usuario Registrado)

Inicio (Mapa)
- Explorar spots (igual)
- Crear spot
  - Formulario
  - Vista previa
- Mi perfil
  - Información personal
  - Mis spots
    - Editar
    - Eliminar
  - Mis favoritos
  - Mis comentarios
  - Configuración
    - Cambiar contraseña
    - Notificaciones
    - Privacidad
- Notificaciones
  - Ver notificaciones
  - Marcar como leída
- Logout

Estructura de Sitio (Moderador)

Panel de Moderación
├── Reportes pendientes
│ ├── Filtrar por tipo
│ ├── Ver detalles
│ └── Tomar acción (aprobar/ocultar/eliminar)
├── Contenido oculto
│ ├── Spots ocultos
│ ├── Comentarios ocultos
│ └── Historial de decisiones
└── Estadísticas
├── Reportes por día/semana
├── Contenido eliminado
└── Usuarios reportados

Estructura de Sitio (Administrador)

Panel de Administración
├── Dashboard
│ ├── Métricas globales
│ ├── Gráficos de actividad
│ └── Últimas acciones
├── Gestión de usuarios
│ ├── Listar usuarios
│ ├── Suspender/Activar
│ ├── Ver reportes de usuario
│ └── Establecer roles
├── Gestión de contenido
│ ├── Categorías (crear/editar/eliminar)
│ ├── Tags (crear/editar/eliminar)
│ ├── Spots (listar, editar, eliminar)
│ └── Comentarios (moderar)
├── Reportes
│ ├── Reportes resueltos
│ ├── Estadísticas
│ └── Exportar datos
├── Configuración
│ ├── Ajustes globales
│ ├── Tema (claro/oscuro)
│ ├── Mantenimiento
│ └── Backups
└── Logs
├── Actividad de usuarios
├── Cambios administrativos
└── Errores del sistema

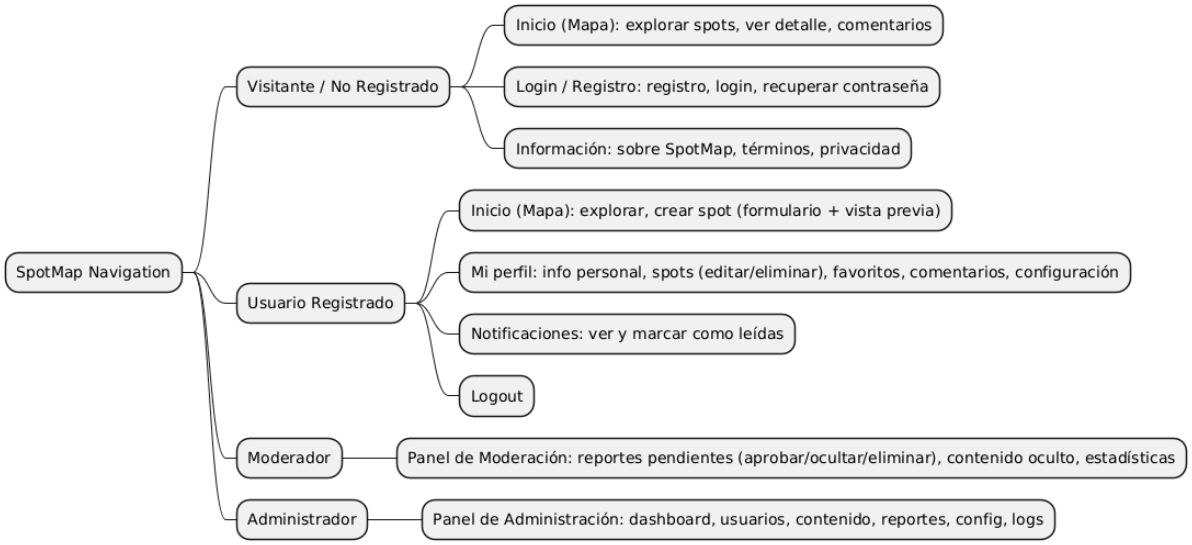


Figura 7.3: Mapa de navegación por roles (visitante, usuario, moderador, administrador) y sus accesos principales.

8. Manual de Usuario

8.1 Manual del Cliente

¿Qué es SpotMap?

SpotMap es una plataforma web colaborativa que centraliza y organiza spots fotográficos mediante geolocalización interactiva. Permite a fotógrafos, viajeros y creadores de contenido descubrir, compartir y valorar lugares fotográficos de interés.

Guía de Inicio Rápido

Paso 1: Crear una Cuenta

1. Accede a **www.spotmap.com**
2. Haz clic en el botón **"Registrarse"** en la esquina superior derecha
3. Completa el formulario:
    - Email válido
    - Contraseña segura (mínimo 8 caracteres, con mayúsculas, números y símbolos)
    - Nombre completo
4. Acepta los términos y condiciones
5. Haz clic en **"Crear cuenta"**
6. Verifica tu email haciendo clic en el enlace recibido

Paso 2: Explorar Spots en el Mapa

1. Desde la pantalla de inicio, visualiza el **mapa interactivo**
2. Usa los **controles de zoom** (+ y -) para acercarte/alejarte
3. Usa los **filtros**:
    - Categoría: elige tipo de spot (mirador, playa, urbano, montaña)
    - Distancia: establece rango de km desde tu ubicación
    - Popularidad: ordena por valoración o recientes
4. Haz clic en cualquier **marcador** para ver detalles del spot

Paso 3: Ver Detalles de un Spot

Cuando haces clic en un marcador, se abre un panel con: - **Galería de imágenes**: desliza para ver todas las fotos - **Información**: descripción, categoría, ubicación exacta - **Estadísticas**: valoración media (estrellas), número de comentarios - **Comentarios**: lee opiniones de otros usuarios - **Botones de acción**: - Añadir a favoritos (corazón) - Valorar (estrellas 1-5) - Reportar (si es inapropiado)

Paso 4: Crear tu Propio Spot

1. Haz clic en el botón **"+ Crear spot"** (flotante en la esquina)
2. Completa el formulario:
    - **Título**: nombre del lugar (ej: "Mirador del Alcázar")
    - **Descripción**: detalles, consejos, horarios (opcional)
    - **Categoría**: selecciona tipo de spot
    - **Ubicación**: busca dirección o haz clic en el mapa
    - **Imágenes**: sube 1-10 fotos (JPG/PNG, máx 5MB cada una)
    - **Etiquetas**: añade palabras clave (ej: #atardecer, #fotografía)
3. Haz clic en **"Vista previa"** para revisar
4. Haz clic en **"Publicar"** para guardar

Paso 5: Comentar y Valorar

En la pantalla de detalle de un spot:

Para valorar: 1. Haz clic en las estrellas (1-5) 2. Se guarda automáticamente 3. Tu valoración cuenta en la media

Para comentar: 1. Desplázate hasta la sección "Comentarios" 2. Escribe tu comentario en el campo de texto 3. Haz clic en **"Enviar"** 4. Para responder a un comentario, haz clic en **"Responder"** debajo del mismo

Mi Perfil

Acceder al Perfil

1. Haz clic en tu **avatar** (esquina superior derecha)

2. Selecciona **“Mi perfil”**

Ver Información Personal

- Avatar
- Nombre completo
- Email
- Fecha de registro
- Número de spots creados
- Número de comentarios

Mis Spots

1. En el perfil, ve a la sección **“Mis spots”**
2. Verás una lista de todos los spots que has creado
3. Acciones disponibles:
    - **Editar**: modifica título, descripción, imágenes
    - **Eliminar**: borra el spot (irreversible)
    - **Ver estadísticas**: visitas, valoraciones, comentarios

Mis Favoritos

1. En el perfil, ve a la sección **“Mis favoritos”**
2. Verás todos los spots que has guardado
3. Haz clic en el corazón para quitar de favoritos

Configuración

1. En el perfil, haz clic en **“Configuración”**
2. Opciones disponibles:
    - **Cambiar contraseña**: introduce contraseña actual y nueva
    - **Notificaciones**: habilita/deshabilita alertas por email
    - **Privacidad**: controla si tu perfil es público/privado
    - **Conectar con redes sociales** (opcional)

Cerrar Sesión

1. Haz clic en tu avatar
2. Selecciona **“Cerrar sesión”**

Funcionalidades Avanzadas

Búsqueda Avanzada

1. En el mapa, usa la **barra de búsqueda** en la parte superior
2. Tipos de búsqueda:
    - Por nombre: “Mirador”
    - Por localidad: “Barcelona”
    - Por etiqueta: “#atardecer”
3. Presiona Enter o haz clic en la lupa

Filtros Combinados

Combina múltiples filtros: 1. Categoría + Distancia: “Playas a menos de 50km” 2. Popularidad + Recientes: “Spots mejor valorados creados hoy” 3. Ubicación + Categoría: “Spots urbanos en Madrid”

Reportar Contenido Inapropiado

Si encuentras contenido que viola las normas: 1. En el detalle del spot/comentario, haz clic en **“⋮ Más opciones”** 2. Selecciona **“Reportar”** 3. Elige motivo: - Spam - Contenido inapropiado - Duplicado - Falso/engañoso - Otro 4. Añade descripción (opcional) 5. Haz clic en **“Enviar reporte”**

Normas de Uso

- **Respeto**: Sé respetuoso en comentarios. No toleramos insultos ni odio.
- **Autenticidad**: Sube solo fotos propias o con permiso del autor.
- **Descripción precisa**: Asegúrate de que la ubicación sea correcta.
- **Sin spam**: No promociones negocios sin relevancia fotográfica.
- **Sin menores**: Respeta la privacidad de menores en fotos.

Preguntas Frecuentes (FAQ)

¿Puedo eliminar mi cuenta?
Sí, desde Configuración → Privacidad → “Eliminar cuenta”. Todos tus datos se borrarán.

¿Qué pasa si alguien reporta mi spot?
Un moderador lo revisará. Si viola las normas, se ocultará; si es incorrecto, se archivará.

¿Puedo editar un comentario?
Sí, puedes editar tus comentarios en los 5 minutos posteriores a su creación.

¿Cómo funcionan los favoritos?
Son privados. Solo tú ves tus favoritos. Se sincronizan en todos tus dispositivos.

¿Hay límite de spots que puedo crear?
No, puedes crear ilimitados spots. Recomendamos calidad sobre cantidad.

8.2 Manual de Instalación y Despliegue

Requisitos Previos

Hardware: - Servidor con mínimo 2GB RAM, 20GB almacenamiento SSD - Procesador con 2 cores - Conexión a internet estable

Software: - Node.js 18+ (para build frontend) - PHP 8.2+ - PostgreSQL 14+ - Git - Composer (gestor de dependencias PHP)

Cuenta en servicios cloud: - Supabase (PostgreSQL + Storage + Auth) - Vercel o similar (deploy frontend) - SendGrid/Mailgun (envío de emails)

Instalación Local (Desarrollo)

1. Clonar Repositorio

```
git clone https://github.com/tuusuario/spotmap.git
cd spotmap
```

2. Configurar Backend (PHP)

```
cd backend

# Instalar dependencias
composer install

# Copiar archivo de configuración
cp .env.example .env

# Generar clave de aplicación
php artisan key:generate

# Configurar variables de entorno
# Edita .env con credenciales de Supabase:
# DB_CONNECTION=pgsql
# DB_HOST=tu-proyecto.supabase.co
# DB_PORT=5432
# DB_DATABASE=postgres
# DB_USERNAME=postgres
# DB_PASSWORD=tu-contraseña
# SUPABASE_URL=https://tu-proyecto.supabase.co
# SUPABASE_KEY=tu-key

# Ejecutar migraciones
php artisan migrate

# Crear usuario admin inicial
php artisan db:seed
```

```
# Iniciar servidor development
php artisan serve
```

Backend disponible en: http://localhost:8000/api

3. Configurar Frontend (JavaScript)

```
cd frontend

# Instalar dependencias
npm install

# Crear archivo de configuración
cp .env.example .env.local

# Editar .env.local:
# VITE_API_URL=http://localhost:8000/api
# VITE_SUPABASE_URL=tu-url-supabase
# VITE_SUPABASE_KEY=tu-key

# Iniciar servidor development
npm run dev
```

Frontend disponible en: http://localhost:5173

4. Verificar Funcionamiento

- Abre el navegador: http://localhost:5173
- Intenta registrar una cuenta
- Crea un spot de prueba
- Verifica que la geolocalización funciona

Despliegue en Producción

Opción 1: Despliegue en Vercel (Frontend + Serverless Backend)

Paso 1: Preparar Repositorio

```
# Asegúrate de que el repo está en GitHub
git remote add origin https://github.com/tuusuario/spotmap.git
git push -u origin main
```

Paso 2: Desplegar Frontend en Vercel

1. Accede a https://vercel.com/
2. Importa el repositorio de GitHub
3. Configura:

- Build command: cd frontend && npm run build
- Output directory: frontend/dist
- Environment variables:

  VITE_API_URL=https://tu-api.com/api
  VITE_SUPABASE_URL=tu-url-supabase
  VITE_SUPABASE_KEY=tu-key

4. Haz clic en Deploy

Frontend disponible en: https://tu-proyecto.vercel.app

Paso 3: Desplegar Backend

Opción A: Vercel Serverless (recomendado)

1. Convierte backend a serverless functions:

```
mkdir -p api

# Crear archivo /api/index.php que ejecute Laravel
echo '<?php
require __DIR__ . "/../backend/public/index.php";
' > api/index.php
```

2. En Vercel, configura:
   - Build command: cd backend && composer install
   - Environment variables: (DB, Supabase, etc.)
3. Deploy

Backend disponible en: https://tu-proyecto.vercel.app/api

Opción B: Servidor dedicado (DigitalOcean, Linode, AWS)

```
# SSH al servidor
ssh root@tu-servidor-ip

# Clonar repositorio
git clone https://github.com/tuusuario/spotmap.git /var/www/spotmap

# Instalar dependencias
cd /var/www/spotmap/backend
composer install --optimize-autoloader --no-dev

# Configurar .env con credenciales producción
nano .env

# Ejecutar migraciones
php artisan migrate --force
```

```
# Configurar Nginx
sudo nano /etc/nginx/sites-available/spotmap

# Contenido mínimo de Nginx:
# server {
#     listen 80;
#     server_name tu-dominio.com;
#
#     root /var/www/spotmap/backend/public;
#
#     location / {
#         try_files $uri $uri/ /index.php?$query_string;
#     }
#
#     location ~ \.php$ {
#         include fastcgi_params;
#         fastcgi_pass unix:/run/php/php8.2-fpm.sock;
#     }
# }

# Habilitar sitio
sudo ln -s /etc/nginx/sites-available/spotmap /etc/nginx/sites-enabled/
sudo nginx -t
sudo systemctl restart nginx

# Configurar SSL con Let's Encrypt
sudo certbot certonly --nginx -d tu-dominio.com

# Reiniciar servicios
sudo systemctl restart php8.2-fpm
sudo systemctl restart nginx
```

Backend disponible en: https://tu-dominio.com/api

Opción 2: Despliegue Completo en Docker

Paso 1: Crear Dockerfile

```
FROM php:8.2-fpm

RUN apt-get update && apt-get install -y \
    libpq-dev \
    git \
    curl \
    unzip

RUN docker-php-ext-install pdo pdo_pgsql
```

```
WORKDIR /app

COPY backend/ /app/

RUN curl -sS https://getcomposer.org/installer | php -- --install-dir=/usr/local/bin --filename=compose
r

RUN composer install --no-dev --optimize-autoloader

EXPOSE 9000

CMD ["php-fpm"]
```

Paso 2: Crear docker-compose.yml

```
version: '3.8'

services:
  php:
    build: .
    container_name: spotmap-api
    environment:
      - DB_HOST=postgres
      - DB_DATABASE=spotmap
      - DB_USERNAME=postgres
      - DB_PASSWORD=tu-contraseña
    depends_on:
      - postgres
    ports:
      - "9000:9000"

  postgres:
    image: postgres:14
    container_name: spotmap-db
    environment:
      POSTGRES_DB: spotmap
      POSTGRES_PASSWORD: tu-contraseña
    volumes:
      - postgres_data:/var/lib/postgresql/data
    ports:
      - "5432:5432"

  nginx:
    image: nginx:alpine
    container_name: spotmap-web
    ports:
      - "80:80"
    volumes:
```

```
      - ./nginx.conf:/etc/nginx/nginx.conf:ro
      - ./backend/public:/app/public:ro
    depends_on:
      - php

volumes:
  postgres_data:
```

Paso 3: Ejecutar Contenedores

```
docker-compose up -d

# Ver logs
docker-compose logs -f php

# Ejecutar migraciones
docker-compose exec php php artisan migrate

# Crear usuario admin
docker-compose exec php php artisan db:seed
```

Configuración de Dominio y SSL

Configurar Dominio

1. En tu registrador de dominios (GoDaddy, Namecheap, etc.):
2. Apunta los registros DNS:

   A record: tu-dominio.com → IP-de-tu-servidor
   CNAME: www.tu-dominio.com → tu-dominio.com

Instalar Certificado SSL

```
# Instalar Certbot
sudo apt-get install certbot python3-certbot-nginx

# Generar certificado (automático)
sudo certbot certonly --nginx -d tu-dominio.com -d www.tu-dominio.com

# Auto-renovación
sudo systemctl enable certbot.timer
sudo systemctl start certbot.timer
```

Monitoreo y Mantenimiento

Logs

```
# Backend (PHP)
tail -f /var/log/nginx/error.log
tail -f /var/log/php8.2-fpm.log

# Frontend (Vercel)
Accede a dashboard de Vercel para ver logs
```

Backups

```
# Backup de base de datos (PostgreSQL)
pg_dump -U postgres spotmap > backup_$(date +%Y-%m-%d).sql

# Restaurar
psql -U postgres spotmap < backup_2025-12-04.sql

# Backup de storage (Supabase)
Realiza backups automáticos desde dashboard de Supabase
```

Actualizar Aplicación

```
# Backend
cd /var/www/spotmap/backend
git pull origin main
composer install
php artisan migrate --force
systemctl restart php8.2-fpm

# Frontend (Vercel automático al push en main)
git push origin main
```

Troubleshooting

Problema	Solución
Error de conexión a BD	Verifica credenciales en .env, conectividad a Supabase
CORS errors en frontend	Configura headers en backend: Access-Control-Allow-Origin
Imágenes no se suben	Verifica permisos en Supabase Storage, límite de tamaño
Mapa no carga	Verifica API key de Leaflet/OpenStreetMap
Notificaciones no llegan	Revisa config de SendGrid/Mailgun, logs de email