

# 1) Arquitectura definitiva (simple, sólida y escalable)

## Capas

1. **WordPress (Backoffice + datos)**
2. **API REST propia (lectura pública)**
3. **Frontend**
  - Fase 1: dentro de WP (plantilla + JS/Leaflet)
  - Fase 2: React/Next consumiendo la misma API

### Regla de oro:

👉 *El mapa nunca habla con WP directamente. Solo con tu API.*

---

## Modelo de datos (WordPress)

### CPTs

#### 1) **br\_story** (Historias)

Unidad mínima. Todo pivota aquí.

#### 2) **br\_route** (Rutas)

Una ruta es un **conjunto ordenado de historias**.

---

### Taxonomías (para filtrar rápido, sin hacks)

Para **br\_story**:

- **br\_district** (Distrito) *(idealmente 1 por historia)*

- `br_neighborhood` (Barrio) *(opcional pero muy útil)*
- `br_period` (Época) *(0..n)*
- `br_theme` (Tema principal) *(0..n)*
- `br_story_type` (Tipo de historia) *(1)*
- `post_tag` o `br_tag` (Etiquetas extra) *(0..n)*

Para `br_route`:

- `br_period` *(0..n)*
- `br_theme` *(0..n)*
- `br_district` *(0..n)*

Distritos y épocas como taxonomías te dan filtros rápidos + URLs limpias + SEO.

---

## Meta fields (campos) — alineado a tu plantilla

Para `br_story`:

- `br_id_internal` (string: "0001")
- `br_subtitle` (string)
- `br_full_address_approx` (string)
- `br_lat` (float)
- `br_lng` (float)
- `br_year_start` (int|null)
- `br_year_end` (int|null)
- `br_reading_time_min` (int|null)

- `br_show_on_map` (bool)
- `br_status` (enum: `pending`|`reviewed`|`approved`)
- `br_source` (string/url)
- `br_author_text` (string)
- `br_internal_notes` (string) (*solo admin*)

Para `br_route`:

- `br_route_stops` (array ordenado de IDs de historias)
- `br_time_estimated_min` (int|null)
- `br_distance_estimated_km` (float|null)
- `br_route_level` (enum: `easy`|`medium`|`hard`) (*opcional*)
- `br_route_start_lat/lng` (*opcional si quieres "punto inicio"*)

---

## Reglas editoriales (anti-caos)

Solo aparece al público si:

- `br_status = approved`
- `br_show_on_map = true`
- `br_lat` y `br_lng` existen

Lo demás se queda en backstage.

---

## 2) API REST (mínima pero perfecta)

Prefijo: `/wp-json/br/v1/`

## Endpoint A — markers del mapa (ligerísimo)

GET /map/stories

### Query params

- bbox=minLng,minLat,maxLng,maxLat (*clave*)
- zoom=12
- q=texto
- district=slug
- period=slug
- theme=slug
- type=slug
- tag=slug
- year\_from=1800
- year\_to=1920
- limit=500

### Respuesta (ejemplo)

```
{
  "count": 2,
  "items": [
    {
      "id": 123,
      "slug": "el-drac-modernista",
      "title": "El drac modernista",
      "subtitle": "Una pieza oculta en la fachada...",
      "lat": 41.387,
      "lng": 2.169,
      "year_start": 1902,
      "year_end": 1902,
    }
  ]
}
```

```
    "district": ["eixample"],
    "neighborhood": ["dreta-de-leixample"],
    "period": ["modernisme"],
    "theme": ["arquitectura"],
    "type": ["anecdota"],
    "status": "approved",
    "reading_time_min": 6
  }
]
```

👉 Este endpoint **no devuelve el texto completo**. Solo lo necesario para pintar y filtrar.

---

## Endpoint B — detalle de historia

GET /stories/{id} o /stories/{slug}

Devuelve:

- contenido completo
  - fuente/autor
  - términos (época/tema/distrito...)
  - recomendadas (misma época/tema/distrito)
- 

## Endpoint C — listado de rutas

GET /routes

- filtros opcionales por época/tema/distrito

## Endpoint D — detalle de ruta

GET /routes/{id}

Devuelve:

- stops ordenados
  - para cada stop: id, title, lat/Ing, año, resumen corto
- 

## Endpoint E — taxonomías (para pintar filtros dinámicos)

- `GET /tax/districts`
- `GET /tax/periods`
- `GET /tax/themes`
- `GET /tax/types`

Esto te permite que el frontend **no tenga filtros hardcodeados**.

---

## 3) Frontend (fase 1 WP / fase 2 React) sin duplicar trabajo

### Contrato único

Tu frontend (sea WP o React) solo necesita:

- `/map/stories` para pintar
- `/stories/{id}` para abrir
- `/routes + /routes/{id}` para rutas
- `/tax/*` para filtros

Así podrás migrar a React/Next cuando quieras **sin tocar WP**.

---

## 4) Rendimiento: reglas simples que lo hacen rápido

1. **Carga por bounds (bbox)**
    - al mover el mapa o cambiar filtros → pides markers del área visible
  2. **Throttle/debounce**
    - no dispares 20 requests seguidas al arrastrar
  3. **Caching**
    - cache del endpoint `/map/stories` por combinación:
      - bbox redondeado + filtros + zoom
    - invalida cache cuando una historia “approved” se actualiza
  4. **Clustering**
    - a partir de X marcadores (p.ej. 200) activa clusters
- 

## 5) Flujo ideal de usuario (UX) — definido de verdad

### Pantalla 1 — Home (Mapa como puerta)

#### Elementos

- Fondo: mapa
- Menú escudo (vertical):  
Mapa / Historias / Ruta / Épocas / Distritos / Contacto
- CTA suave arriba o en drawer:
  - “Explorar cerca de mí”

- “Ver rutas destacadas”

## Comportamiento

- Al entrar: ya hay marcadores del área inicial (Barcelona centro o última vista guardada)
  - Panel filtros compacto (chips)
- 

## Flujo A — Explorar (modo principal)

1. Usuario mueve mapa o usa chips (Época/Distrito/Tema)
2. Click en marcador
3. Se abre **StoryDrawer** (panel lateral)
  - título + subtítulo
  - mini resumen (2–4 líneas)
  - datos: época, años, distrito
  - botones:
    - “Leer completa”
    - “Ver cerca”
    - “Historias relacionadas”
4. “Leer completa” abre modo lectura limpio
5. Desde lectura:
  - “Volver al mapa”
  - “Siguiendo cercana”
  - “Ruta relacionada”

**Objetivo:** que encadene historias sin esfuerzo.



---

## Flujo B — Rutas (modo intención)

1. Usuario entra en “Ruta”
2. Ve cards con:
  - nombre, duración, paradas, tema/época, nivel
3. Entra a una ruta:
  - mapa con paradas numeradas
  - botón grande: “Siguiendo parada”
4. Cada parada:
  - preview + “leer completa”
  - botón “volver a ruta” siempre visible
5. Final:
  - resumen + “Rutas similares” + “Compartir”

---

## Flujo C — Épocas / Distritos (modo puerta narrativa)

Página de época/distrito:

- mini introducción (10–12 líneas)
- 6 historias destacadas
- botón: “Ver todo en mapa” (abre mapa con filtros ya aplicados)

---

## 6) Plan de implementación (orden correcto, sin caos)

## Semana 1 (base)

1. Registrar CPTs + taxonomías
2. Registrar meta fields
3. Crear pantalla admin usable (ACF o metabox)
4. Aplicar reglas: solo approved + show\_on\_map salen

## Semana 2 (API)

5. Endpoint `/map/stories` (ligero + filtros + bbox)
6. Endpoint `/stories/{id}`
7. Endpoints `/routes` y `/routes/{id}`
8. Endpoints `/tax/*`

## Semana 3 (UI mapa)

9. Leaflet: pintar markers desde API
10. filtros → refrescan markers
11. drawer de historia

## Semana 4 (rutas + polish)

12. rutas con paradas ordenadas
13. clustering + caching
14. páginas de época/distrito