Auditoría Post-Migración Angular 17 – PokedexApp

Fecha: 29 de octubre de 2025
Responsable: GitHub Copilot (Asistente de Arquitectura Angular)
Alcance: Revisión técnica posterior al salto controlado de Angular 16.2.x a Angular 17.3.x y adopción inicial de Signals.

---

1. Resumen Ejecutivo

- La plataforma compila con dependencias centrales actualizadas a Angular **17.3.12** y CLI **17.3.17**, con TypeScript 5.4.5 y Zone.js 0.14.10.
- Se introdujeron `signals`, `computed` y `effect` en servicios y componentes clave (`PokedexCrudService`, `PokedexService`, `HomeComponent`, `ListComponent`, `CardPokemonDetailsComponent`, `PokedexComponent`).
- Varias vistas migraron a `ChangeDetectionStrategy.OnPush`, reduciendo el trabajo del detector de cambios.
- Persisten áreas vulnerables: enrutamiento sin vista 404 real, uso intensivo de `MatTableDataSource` con suscripciones manuales, carencia de `HttpInterceptor` y manejo directo de `localStorage` sin abstracciones.
- El `environment.development.ts` mantiene `production: true`, configuración riesgosa para ciclos de desarrollo y depuración.
- No se registran nuevas pruebas automatizadas ni ajustes en pipelines, por lo que se requiere validar regresiones antes de avanzar a Angular 18+.

---

2. Información de Base

- Rama analizada: `main` (ahead 2 of origin).
- Archivos modificados principales: `package.json`, `package-lock.json`, servicios de Pokedex, páginas `home/list/pokedex`, componentes `card-pokemon` y `card-pokemon-details`.
- Fuentes consultadas: código fuente actual, documentos `GEMINI.md` y `AUDIT_REPORT.md`.

---

3. Actualización Técnica

3.1 Dependencias

Paquete	Versión anterior	Versión actual	Comentario
`@angular/*` núcleo	`^16.2.0`	`^17.3.12`	Migración exitosa; no se activó modo standalone global.
`@angular/material` / `@angular/cdk`	`^16.2.x`	`^17.3.x`	Sin breaking changes relevantes detectados.

Paquete	Versión anterior	Versión actual	Comentario
`@angular-devkit/build-angular` / `@angular/cli`	`^16.2.10`	`^17.3.17`	Requiere regenerar archivos de configuración si se habilitan builders nuevos.
`zone.js`	`~0.13.0`	`~0.14.10`	Alineado con requirements de Angular 17.
`typescript`	`~5.1.3`	`~5.4.5`	Abrir paso a decoradores TC39 y mejoras de type safety.
`rxjs`	`~7.8.0`	`~7.8.0`	Sin cambios; considerar plan para RxJS 8 en Angular 18.

3.2 Configuración de Build/Test

- Scripts de `package.json` sin nuevas tareas (`ng test`, `ng build`).
- No hay ajustes en `tsconfig*.json`; aún no se activan flags estrictos (`strictTemplates`, `noImplicitOverride`, etc.).
- No se actualizó configuración de linting; continua la ausencia de ESLint en el flujo oficial.

---

4. Arquitectura y Modularización

4.1 Estructura

- Se mantiene `AppModule` mínimo que delega en `PokemonsModule` mediante lazy loading.
- `PokemonsModule` agrupa páginas, componentes y servicios Dominios/Infra mezclados; conviene comenzar la separación `CoreModule`/`SharedModule` o migrar gradualmente a componentes standalone.

4.2 Enrutamiento

- `AppRoutingModule` redirige `**` a `404`, pero no existe vista ni módulo `NotFound`; navegar a rutas inexistentes genera fallback defectuoso.
- No hay guards ni resolvers; recomendable introducir `canMatch`/`canActivate` cuando aparezca autenticación.

4.3 Componentes y Detección de Cambios

- `CardPokemonComponent`, `ListComponent` y `CardPokemonDetailsComponent` adoptan `OnPush`, pero `HomeComponent` y `PokedexComponent` permanecen con `Default`.
- Con signals, ambos componentes restantes pueden migrar a `OnPush` sin fricción.

4.4 Gestión de Estado

- `PokedexCrudService` sustituye `BehaviorSubject` por `signal` + `computed`; operaciones `setFavoritePokemon`/`removeFavoritePokemon` actualizan el estado y persisten en `localStorage`.
- `PokedexService` expone `pokemonDetails` como `signal` y la sincroniza en `getPokemonDetails$()` con `tap`.

- `ListComponent` crea un `effect` que proyecta `pokedex()` sobre `MatTableDataSource`. Aunque funcional, mezcla paradigmas (signals + estructuras imperativas) y mantiene subscripciones manuales (`subscribe()` con `takeUntil`).
- Recomendación: encapsular la sincronización en una *facade* o adaptar la tabla a `dataSource = computed(...)` con `MatTableModule` standalone.

4.5 Servicios e Inyección

- Servicios continúan con `providedIn: 'root'`.
- `MessageSnackbarService` todavía exige que el consumidor provea `MatSnackBar`; migrar a inyección directa simplificaría el API.
- Sin `HttpInterceptor` para gestión de errores, ni servicio de logging central; permanecen múltiples `console.log`/`console.warn`.

---

5. Calidad, Seguridad y Observabilidad

- `localStorage` se usa directamente; sugerido encapsularlo detrás de un adapter (manejo de JSON inválido, expiración, SSR).
- `environment.development.ts` conserva `production: true`; ajustarlo para evitar optimizaciones de prod en entorno local.
- Sigue sin aplicarse `DomSanitizer` (aunque no se utiliza `innerHTML` dinámico).
- No hay interceptores para retiros de token o captura de errores HTTP.
- Pipeline de pruebas sin cambios; necesario ejecutar `ng test --no-watch --code-coverage` tras la migración para detectar regresiones.
- No se registran métricas de rendimiento ni análisis de bundle post-actualización.

---

6. Riesgos Detectados

1. **Fallo UX ante rutas inválidas:** Ruta comodín redirige a `404` inexistente (`src/app/app-routing.module.ts`).
2. **Divergencia Signals ↔ RxJS:** `ListComponent` combina subscriptions imperativas y `effect`; riesgo de desincronización si se introduce paginación server-side.
3. **Persistencia frágil:** Mutaciones sobre arrays clonados en `removeFavoritePokemon()` podrían derivar en inconsistencias si se amplía el estado; preferir `filter`.
4. **Configuración de entorno incorrecta:** `production: true` en desarrollo sigue siendo punto débil heredado de la auditoría previa.
5. **Observabilidad limitada:** Continúan `console.log` dispersos sin nivel de severidad ni mecanismo de trazas centralizado.

---

7. Recomendaciones Inmediatas

1. **Corregir entornos:** Ajustar `environment.development.ts` a `production: false` y documentar implicaciones.
2. **Completar `OnPush`:** Migrar `HomeComponent` y `PokedexComponent` a `ChangeDetectionStrategy.OnPush` aprovechando el respaldo de signals.

3. **Refactorizar sincronización de tabla:** Sustituir suscripciones manuales en `ListComponent` por `async` pipe o `computed` que alimente `MatTableDataSource`.
4. **Implementar `NotFoundComponent`:** Crear módulo/página 404 real y actualizar `AppRoutingModule`.
5. **Estandarizar persistencia:** Encapsular `localStorage` en un `StorageService` inyectable que maneje parseos y errores.
6. **Ejecutar pruebas y cobertura:** Correr `ng test --no-watch --code-coverage` y registrar métricas antes de continuar la migración incremental.
7. **Plan de logging:** Introducir servicio de logging (infraestructura) con niveles y posibilidad de redirigir a observabilidad externa.

---

8. Ruta Hacia Angular 18 → 20

1. **Angular 18:**
   - Ejecutar `ng update @angular/core@18 @angular/cli@18`.
   - Adoptar `@angular/material` 18 y validar breaking changes en theming.
   - Migrar RxJS a v8 (`rxjs-compat` si es preciso) y actualizar `eslint`/`tsconfig` para signals-first patterns.
2. **Angular 19:**
   - Evaluar transición a componentes standalone y `provideRouter`.
   - Introducir `Deferred Loading` (`@for`, `@if`) para listas grandes y mejoras de rendimiento.
   - Reforzar tipado estricto (`angularCompilerOptions.strictTemplates = true`).
3. **Angular 20:**
   - Consolidar signals en toda la capa de presentación, eliminar `NgModule` residuales si se adopta standalone.
   - Alinear bundler (esbuild/Vite) y consideraciones de SSR o Hydration si se decide escalar la app.
   - Configurar automatismos de auditoría de dependencias y pipelines CI/CD con checks obligatorios.

---

9. Próximos Entregables

- Documentar en `GEMINI.md` la estrategia incremental y criterios de aceptación por versión.
- Preparar historias de usuario técnicas para: `NotFound`, refactor de persistencia, logging y cobertura mínima.
- Tras cerrar pendientes, generar commit con tag correspondiente a la release Angular 17 (`v17.0.0-incremental` sugerido) y publicar en remoto.