Servicio Nacional de Aprendizaje – SENA

Evidencia de Desempeño: Verificación de procedimientos para la definición de componentes front-end de la aplicación (Listas de chequeo) GA7-220501096-AA4-EV02

Diseño de sitio web y/o móvil

Análisis y desarrollo de software

Ana del pilar Bermeo Lozano

Fecha: 12 de Julio de 2025

Introducción

El presente documento describe la propuesta de migración del sistema web Parkin P.B. desde una arquitectura estática basada en HTML, CSS y JavaScript hacia una implementación en React JS.

La nueva estructura está diseñada bajo un enfoque de componentes reutilizables, modulares y escalables, que permitan mantener la coherencia visual y optimizar el rendimiento de la aplicación.

Cada componente frontend identificado corresponde a una funcionalidad específica del sistema actual —como la gestión de usuarios, vehículos, zonas, accesos y estadísticas—, asegurando que la interfaz se mantenga intuitiva, dinámica y adaptable a distintos dispositivos.

La adopción de React JS busca mejorar la experiencia de usuario, simplificar el mantenimiento del código y facilitar la integración con el backend existente desarrollado en PHP y MySQL.

Objetivo

Diseñar y documentar la arquitectura de componentes frontend para la implementación del sistema Parkin P.B. en React JS, detallando la funcionalidad y justificación de cada componente.

El objetivo es garantizar que la migración preserve las características actuales del software, potencie la escalabilidad y mantenibilidad del proyecto, y permita un desarrollo más ágil y eficiente, adaptado a las necesidades de gestión de un parqueadero inteligente.

1. Estructura General en React

El sistema se migrará a **React JS** para aprovechar su arquitectura basada en componentes reutilizables, el manejo eficiente del estado, y la facilidad para integrar peticiones asíncronas al Backend en PHP/MySQL.
Se utilizará **React Router** para la navegación entre vistas (páginas) y **Hooks** para la gestión del estado y ciclo de vida.

2. Componentes Principales

Componente	Descripción	Justificación
App	Componente raíz que orquesta la estructura, rutas y contextos globales (autenticación, usuario, configuración).	Centraliza la aplicación y evita repetición de código en la configuración global.
Header	Barra de navegación con logotipo, menús y botones de autenticación.	Mantiene coherencia visual y se reutiliza en todas las vistas.
Footer	Pie de página con enlaces y redes sociales.	Componente estático reutilizable en todas las páginas.
HomePage	Versión React de index.html con secciones: Hero, Servicios, Nosotros, Ubicación.	Facilita separar cada sección como subcomponentes y optimiza mantenimiento.
AuthPage	Contendrá **LoginForm** y **RegisterForm** como subcomponentes con validación en tiempo real.	Modulariza el acceso y registro de usuarios; facilita cambios en formularios.
LoginForm	Formulario de inicio de sesión con manejo de estado local y conexión a backend.	Se reutiliza en AuthPage y mantiene la lógica encapsulada.
RegisterForm	Formulario de registro con validación, selector de tipo de usuario y barra de fuerza de contraseña.	Mejora la experiencia de usuario y permite validaciones antes del envío.
UserDashboard	Versión React de panel.html, con tarjetas y tablas para estadísticas, vehículos, historial y zonas.	Componentización permite recargar solo partes específicas sin refrescar toda la página.
AdminDashboard	Versión React de admin.html con pestañas para gestión de usuarios, vehículos, zonas, espacios, cámaras, accesos y transacciones.	Favorece la carga dinámica de datos y reutilización de componentes de tabla.

StatsCard	Tarjeta individual de estadísticas (vehículos, zonas, ingresos, etc.).	Reutilizable en panel de usuario y panel de administrador.
DataTable	Tabla reutilizable con encabezados dinámicos y opciones (buscar, filtrar, exportar).	Evita duplicar código de tablas en distintos módulos.
Modal	Ventana emergente para crear, editar o eliminar elementos (usuarios, vehículos, zonas, etc.).	Aporta consistencia visual y lógica unificada de formularios.
MapComponent	Componente para mostrar el mapa de ubicación (integración futura con Google Maps o Leaflet).	Encapsula la lógica de mapas sin afectar el resto de la aplicación.
ProtectedRoute	Componente que protege rutas y redirige al login si el usuario no está autenticado.	Mejora la seguridad y el control de acceso.

3. Justificación Global

- **Reutilización**: Al separar en componentes pequeños, se evita la duplicación de código y se mejora la mantenibilidad.
- **Escalabilidad**: Se pueden agregar nuevas funcionalidades sin romper la estructura existente.
- **Mantenibilidad**: Los cambios visuales o funcionales se aplican en un solo lugar y afectan a todos los lugares donde se use el componente.
- **Performance**: React renderiza solo los elementos que cambian, mejorando la experiencia del usuario.
- **Integración**: Facilita la conexión con el backend actual en PHP y la futura migración a APIs más modernas.