**Proyecto: Plataforma de Gestión de Incidentes y Soporte Técnico**

**Descripción**: Crea una plataforma web para que los empleados de una empresa de software puedan reportar, gestionar y hacer seguimiento de incidentes técnicos. La plataforma permite al equipo de soporte técnico gestionar los tickets de incidentes, priorizarlos y resolverlos.

**Características clave:**

1. **Autenticación y Autorización (JWT)**:
   * Los usuarios (empleados y técnicos de soporte) deben iniciar sesión. Usa **JWT** para gestionar la autenticación.
   * Diferentes roles: administrador, empleado, y técnico de soporte. Cada rol tiene diferentes permisos y vistas.
2. **Gestión de Tickets de Incidentes**:
   * **CRUD** de tickets: crear, actualizar, eliminar y ver detalles.
   * Cada ticket tiene un estado: abierto, en progreso, resuelto, cerrado.
   * Los técnicos pueden asignarse tickets y dejar notas de progreso.
3. **Notificaciones en Tiempo Real**:
   * Implementa notificaciones para actualizar a los usuarios sobre el progreso de los tickets (puedes usar WebSockets o servicios como Firebase para el backend).
4. **Dashboard de Estadísticas**:
   * Un dashboard que muestra gráficos y estadísticas en tiempo real sobre los tickets:
     + Cantidad de tickets abiertos, resueltos, tiempos de respuesta.
     + Uso de **Chart.js** o **Recharts** para la visualización.
5. **Buscador y Filtros Avanzados**:
   * Implementa un buscador avanzado y filtros por estado, prioridad, o asignación para navegar fácilmente entre los tickets.
6. **Interfaz de Usuario Avanzada**:
   * Usa **Material-UI** o **TailwindCSS** para diseñar una interfaz intuitiva y moderna.
   * Integrar **dark mode** y **light mode** con un toggle.
7. **Componentización y Optimización**:
   * Crea componentes reutilizables (inputs, modales, tablas).
   * Utiliza **React Hooks** avanzados (como useReducer, useContext o useMemo) y **React Query** para optimización de llamadas API.

**Tecnologías sugeridas:**

* **React.js** (con Hooks)
* **React Router** (para navegación)
* **Redux** o **Context API** (para manejar el estado global)
* **Axios** (para peticiones HTTP)
* **Material-UI** o **TailwindCSS**
* **Chart.js** o **Recharts** (para visualizaciones)
* **React Query** (para manejo de datos asíncronos y optimización)

Este proyecto te permitirá manejar una variedad de temas avanzados como gestión de estado, optimización de rendimiento, patrones de diseño y experiencias reales de usuario, todo en un entorno corporativo típico.