Descripción del Proyecto

El proyecto consiste en la creación de un medidor de ocupación para un local, en el cual podrás visualizar cómo varía la ocupación a lo largo del día, mostrando la carga de personas por hora para cada día de la semana. Este sistema es útil para lugares con alta afluencia de público, como gimnasios, teatros, o cualquier espacio con un flujo constante de personas. Los datos de ocupación se simulan como si provinieran de un local real, pero en lugar de obtener datos en vivo, se controlan manualmente a través de una API RESTful.

En este sistema, cada día de la semana estará representado por una barra vertical que muestra el porcentaje de ocupación por cada hora del día. Las barras estarán coloreadas para indicar diferentes niveles de ocupación: verde para baja ocupación, amarillo para media ocupación, y rojo para alta ocupación. Los usuarios podrán interactuar con las barras y obtener detalles sobre la ocupación de cualquier hora del día.

La API permitirá que se agreguen, modifiquen o eliminen datos de ocupación manualmente, lo que simula cómo funcionaría en un escenario real con un sistema de sensores o gestión de reservas.

Herramientas y Cómo Usarlas

1. React (Frontend)
   * Uso: React se utilizará para construir la interfaz de usuario dinámica y responsiva del medidor de ocupación. Aquí se gestionará la visualización en tiempo real de los datos de ocupación.
   * Cómo usarlo:
     + Crear un componente OccupationBar para representar las barras de cada día de la semana, donde se muestre la ocupación por hora.
     + Utilizar el estado de React para gestionar los datos de ocupación (porcentaje de ocupación) que provienen de la API.
     + Usar useEffect para hacer peticiones a la API y obtener los datos de ocupación cuando el componente se renderiza.
     + Utilizar React para actualizar la interfaz dinámicamente según los datos de la API.
2. Tailwind CSS (Estilo)
   * Uso: Tailwind CSS será utilizado para darle estilo a la interfaz de forma rápida y eficiente, proporcionando una experiencia visual atractiva sin tener que escribir mucho CSS personalizado.
   * Cómo usarlo:
     + Utilizar las clases de Tailwind para crear un diseño limpio y moderno, enfocado en la visualización de las barras de ocupación.
     + Personalizar las barras de ocupación utilizando gradientes de color para reflejar los diferentes niveles de ocupación (verde, amarillo, rojo).
     + Implementar clases de diseño responsivo para garantizar que la interfaz se adapte bien a diferentes tamaños de pantalla, garantizando accesibilidad.
3. Firebase (Backend/DB)
   * Uso: Firebase será utilizado para manejar tanto el backend como la base de datos en tiempo real. Firebase proporcionará la funcionalidad de una API RESTful para gestionar los datos de ocupación y almacenarlos en su base de datos Firestore.
   * Cómo usarlo:
     + Utilizar Firestore de Firebase para almacenar los datos de ocupación, lo que permite la persistencia de los mismos.
     + Usar las funciones de Firebase para crear, leer, actualizar y eliminar los datos de ocupación.
     + Configurar la autenticación de Firebase (si es necesario) para controlar quién puede modificar los datos de ocupación.
4. JavaScript (AJAX / Fetch)
   * Uso: JavaScript será utilizado para realizar las solicitudes a la API de Firebase desde el frontend en React.
   * Cómo usarlo:
     + Utilizar fetch o Axios en React para hacer peticiones HTTP GET, POST, PUT, DELETE a la API de Firebase.
     + Los datos de ocupación serán recuperados de Firebase y se actualizará dinámicamente la UI cada vez que haya cambios.
     + Usar fetch para enviar las modificaciones a los datos de ocupación a Firebase cuando se realicen cambios a través de la interfaz de usuario.
5. HTML/CSS
   * Uso: El uso de HTML y CSS se limitará a la estructura básica de la página y los estilos que no estén cubiertos por Tailwind.
   * Cómo usarlo:
     + Crear la estructura básica de la página utilizando HTML.
     + Personalizar algunos estilos mínimos, como la disposición de los elementos, en caso de que Tailwind no cubra todas las necesidades.

Flujo del Proyecto

1. Frontend:
   * En el frontend, React renderiza la interfaz de usuario del medidor de ocupación.
   * El componente OccupationBar representa las barras de ocupación de cada día y las actualiza dinámicamente según los datos obtenidos de Firebase.
   * Se utilizarán peticiones fetch o Axios para obtener y enviar datos a Firebase en tiempo real.
2. Backend/API:
   * Firebase actúa como el backend, proporcionando las funciones de base de datos en tiempo real y una API RESTful para interactuar con los datos de ocupación.
   * Los datos se almacenan en Firestore, permitiendo que el frontend se comunique con Firebase para recuperar, modificar y eliminar información.
3. Comunicación:
   * La comunicación entre el frontend y el backend se realiza mediante solicitudes HTTP usando fetch para enviar y recibir datos.
   * El frontend está vinculado a Firebase, lo que permite que las actualizaciones en los datos de ocupación se reflejen en tiempo real en la interfaz de usuario.

Características y Funcionalidades

* Visualización interactiva: Las barras verticales representan el nivel de ocupación para cada hora de cada día de la semana, con coloración que varía según el nivel de ocupación (verde, amarillo, rojo).
* Control de datos: Los datos de ocupación son controlados a través de Firebase, permitiendo la adición, modificación y eliminación de estos de manera sencilla.
* API RESTful: Firebase maneja las solicitudes de la API para crear, leer, actualizar y eliminar los datos de ocupación.
* Escalabilidad: El sistema puede crecer en el futuro para manejar más datos y realizar análisis más complejos de ocupación.

Posibles Mejoras Futuras

* Integración con sensores reales: Se podría integrar este sistema con un sistema de sensores reales o con datos en vivo, permitiendo la actualización automática de la ocupación en lugar de la entrada manual de datos.
* Interactividad avanzada: Añadir notificaciones en tiempo real que alerten a los usuarios cuando la ocupación de un área específica supera un umbral determinado. Por ejemplo, notificar a los usuarios de un gimnasio si la ocupación es muy alta.
* Autenticación y Roles de Usuario: Implementar un sistema de autenticación de usuarios mediante Firebase Authentication para gestionar roles de acceso y control sobre qué usuarios pueden modificar los datos de ocupación.
* Mejoras en el análisis de datos: En el futuro, se podrían agregar funcionalidades de análisis para predecir picos de ocupación en función de los datos históricos.