TP N°3: Sistema de Reservas

Consigna: vamos a programar una API en Node para manejar las reservas de un cine que solamente proyecta películas de culto. A pesar de proyectar una función cada tanto, la API tiene mucha demanda, por lo cual es importante que pueda manejar múltiples conexiones en paralelo.

Las rutas que tiene que poder atender son las siguientes:

- GET /funciones: devuelve el listado de las funciones **disponibles.** Si ya pasó la fecha de proyección o si las entradas están agotadas, esa función no debería aparecer en el listado.
- POST id_funcion/reservar: recibe en el cuerpo del request el user_id y las butacas a reservar. Si el usuario no reservó previamente entradas para esa función y si está reservando menos de 6 butacas, entonces se genera la reserva. Obviamente no se puede reservar entradas para una función que no está disponible. En caso de reservar las últimas butacas disponibles, marcar la función como no-disponible.
- POST id_funcion/cancelar_reserva: recibe el user_id y anula la reserva del usuario, si falta al menos una hora para la función.

Además, deben configurar una tarea “cron” que corra cada 5 minutos y se fije si una función está por comenzar para marcarla como no-disponible.

Extensión para grupos de 3: generar una versión “naive” del código sin clusterizar y sin transacciones y comparar los tiempos de ejecución para las rutas **funciones** y **reservar**. Pueden usar como base el script *donar.js* de la clase y probar variar la cantidad de requests (ej 10,

100, 1000, etc) para cada ruta de manera que se vea si efectivamente existe una cantidad de requests que hagan que valga la pena clusterizar el código.

Cómo herramienta si quieren generar algunos gráficos y no volverse locos escribiendo código para los experimentos les recomiendo Jupyter Notebook, es una herramienta que les permite correr código python y mostrar resultados de manera muy sencilla.

- Links útiles:
  - Paquete cron de Node
  - Obtener fecha y hora actual en Javascript