**DESARROLLO DE LA PRUEBA**

1. **Registrar las variables globales a utilizar:**

Se crearon 4 variables que muestran datos aleatorios, tres de esas cuatro variables son de tipo CSV Data Set Config, las cuales ameritan de un archivo CSV con datos para alimentar la variable, dichos archivos están alojados en la carpeta “Recurso” indicada en el repositorio, dichas variables almacenan los datos de los campos de crear una orden; Corteza, Sabor y Tamaño. La ultima variable es de tipo Variable Random, ya que son datos numéricos que representan la mesa a la que esta asociada la orden, se configuro la variable para que use de forma aleatoria números comprendidos desde el 0 hasta el 50 por cada iteración información de mesa se creo usando un parámetro false en el campo “Per thread User” con la finalidad que los datos no se repitan por cada iteración

NOTA: En la rama principal del TEST PLAN llamado PIZZA ORDERS se agrego una variable global para almacenar el dominio de la api, se configuro la variable “host” para el dominio “order-pizza-api.herokuapp.com” de esta forma en los Request en el campo “Server Name o IP” se utilizaría la variable ${host}

1. **Agregar Primer Grupo de hilo a ejecutar por usuario:**

Se configuro el Grupo de hilo para ejecutar una simulación de 100 Usuarios por petición en intervalos de 0.33sg Cada uno lo que da un total de 5 min en total para los 100 usuarios para así cumplir con el requerimiento de la prueba, posteriormente se procedió a configurar los “Request” necesarios para este primer grupo:

* 1. GN 1.0 HTTP Header Manager: Se agrego un elemento de configuración que permitiera poder configurar dos datos; El primero es que el contenido enviado en el header en de tipo json, y el segundo para autorizar la variable del token Access para que se pueda utilizar como dato para autorizar la creación de órdenes y no generar un error 401
  2. GN 1.1 Autenticando Usuario: Se agrego un HTTP Request con el método post para generar el token Access para el usuario en la simulación, los datos ingresados son una petición de tipo Post, en el domino ${host}, bajo la ruta /api/auth, donde se ingreso en el “Body Data” los datos de acceso del usuario en este caso fueron Username: test y Password: test
     1. GN 1.1.1 Guardado de token en variable: Una vez enviada la petición, se procedió a agregar un JSON Extractor, el cual permite sustraer el token del Json y almacenarlo en una variable que anteriormente se había configurado en el Header manager para su uso en la peticiones de registrar ordenes nuevas
  3. GN 1.2 Generando nueva orden: Esta es una HTTP Request que se agrego para poder enviar una petición de tipo post, donde vamos a enviar el token previamente guardado mas unos datos adicionales a través de un json con la información complementaria para realizar una orden nueva, esos datos los sustraemos de las variables globales realizadas al comienzo de la automatización, donde se utilizarían los datos, “Crust, Flavor, Table y Size”
  4. GN 1.3 Obtener listado de ordenes: Aquí se agrego un HTTP Request de tipo get, para obtener el listado de ordenes creadas, se configuro con el dominio con el ${host} la ruta enviada fue /api/orders
     1. GN 1.3.1 Extraer el OrderID Aleatorio: Se agrego un JSON Extractor dentro de la petición http para obtener un un Id Random de las ordenes y posteriormente eliminarla en le siguiente paso
  5. GN 1.4 Eliminar una Orden Aleatoria: Por ultimo en este grupo de hilo se agrego un último HTTP Request de tipo DELETE que vaya alminar una orden aleatoria almacenada en una variable en el paso anterior, para esto se configuro el dominio ${host}, la ruta enviada fue /api/orders/{OrderID} donde la variable representa el ID que se eliminara de forma aleatoria

1. **Agregar un segundo grupo de hilo con otra configuración:**

Se agrego un nuevo grupo de hilo, este cuenta con una configuración de 10 registros en intervalos de 1 segundo cada uno dando un total de 10 peticiones en 10 segundos

Este Grupo de hilo posee la misma estructura que el grupo anterior a diferencia que las variables para el token y para el Order ID cambian para no tener conflictos en la ejecución de la prueba anterior

1. **Listener para ver resultados de las pruebas:**

Se agregaron 3 listener para ver los resultados de las pruebas

El primero es de tipo Ver resultado en árbol, que permite visualizar las peticiones una a una mostrando su código de respuesta http y el tiempo de ejecución, al igual que permite visualizar el Request y response Data obtenido en cada iteración

El segundo es de tipo Ver resultado en tabla, mostrara los datos globales de la prueba por cada iteración el tiempo de ejecución, el status de el Request si fue exitoso o genero un error, el promedio en tiempo por cada petición e iteración, la velocidad de repuesta del servidor entre otros datos

Por último se agrego un Summary report, el cual mostrara un reporte el cual se exporto en formato CSV y esta adjunto en le repositorio con los datos obtenidos por la ejecución de cada Request, la cantidad de peticiones que tuvieron cada Request, el tiempo global y una información mas general en cada iteración, entre ellos el porcentaje de error en cada Request, el porcentaje de datos enviados y recibidos por parte del servidor