**Performance:**

Para que la prueba sea realista consultaría con el equipo de producto las distintas métricas con las cuales se cuentan de eventos anteriores, y tambien sobre que lapso de tiempo se espera responder a cierto requerimiento.

Tambien en algunos casos es posible crear escenarios con el uso de caché lo cual tendrá un impacto en el output de la prueba.

TARGET 500 TPS => (500 TPS x 60 S) => 30.000 RPM

Tal vez siga siendo poco pero dividiría la ejecución en un master y 6 nodos en donde enviaría 5k RPM a cada nodo.

500 TPS / 6 Nodos = 84 TPS/N => (84 TPS x 60 S) => ~5.000 RPM

Threads: 840

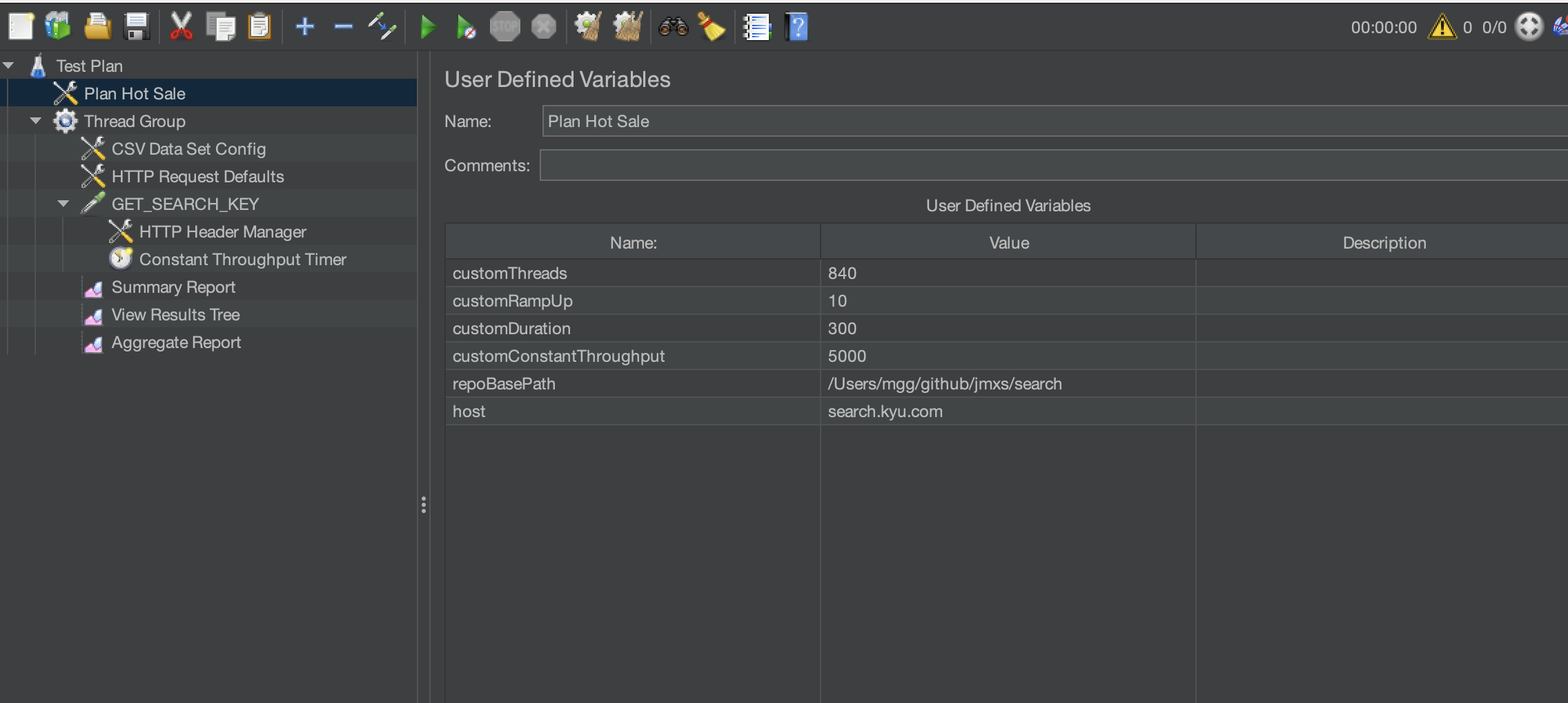
RampUp: 10

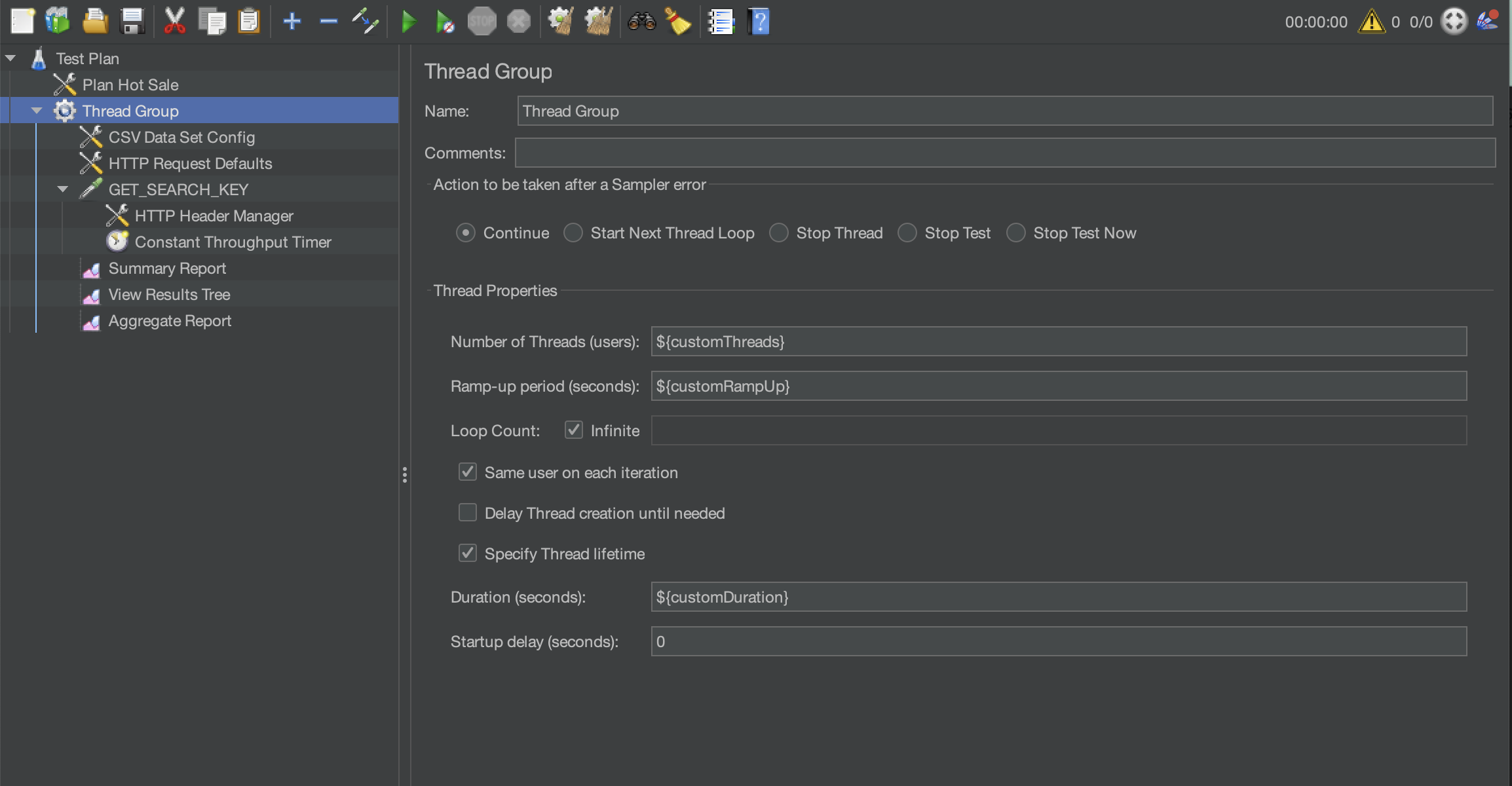
840 (Th) /10 (Rup) = 84 => 84 \* 60s (1 min) => 5.000 RPM

Constant Throughput = 5000

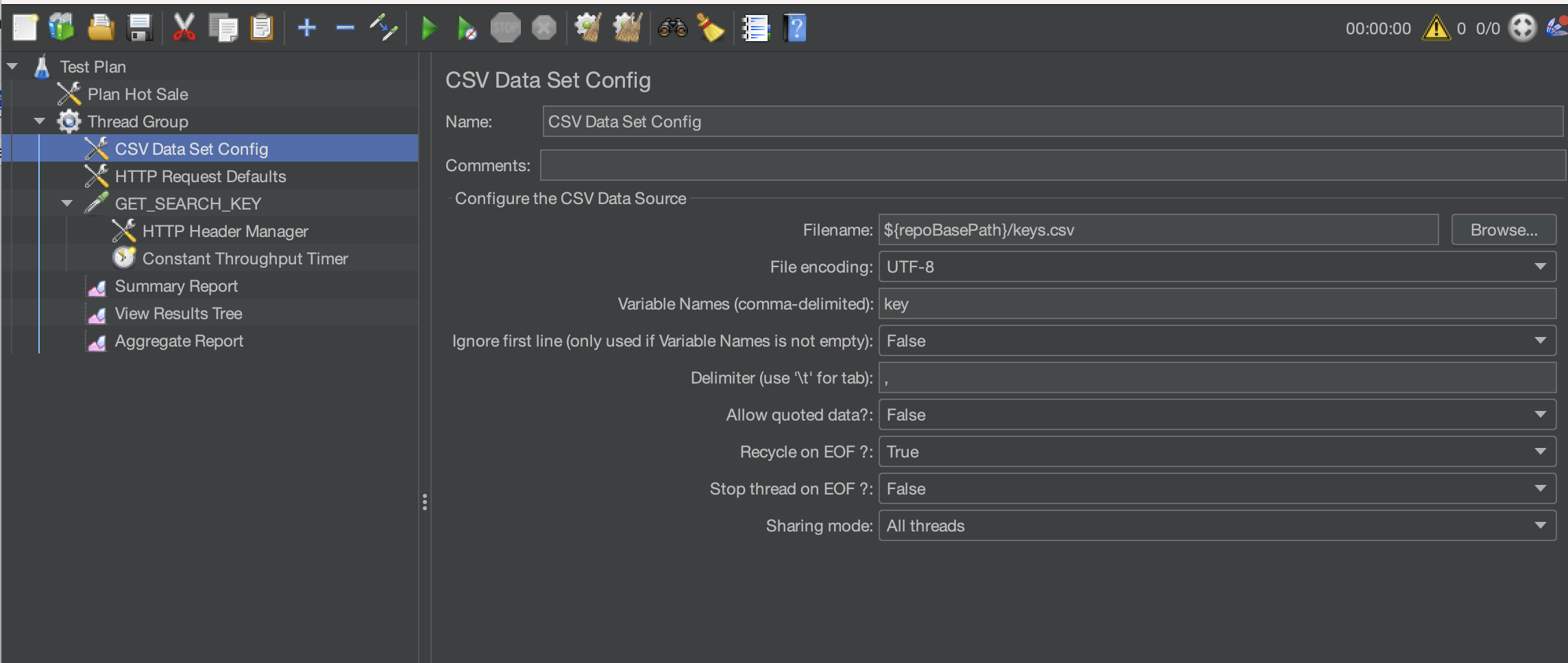
Duration: 300s (5 mins)

-Para que el script pueda ser reutilizable intentariá hacer uso de definición de variables y consumirlas desde los distintos componentes del plan:

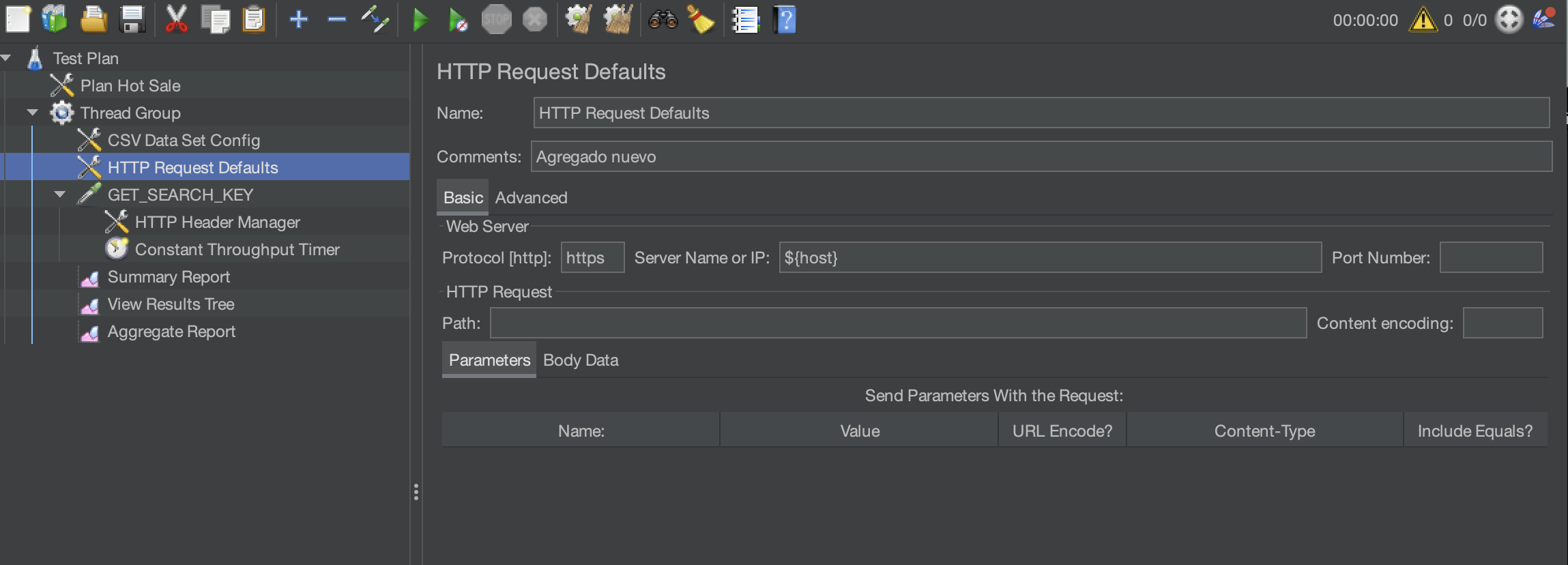




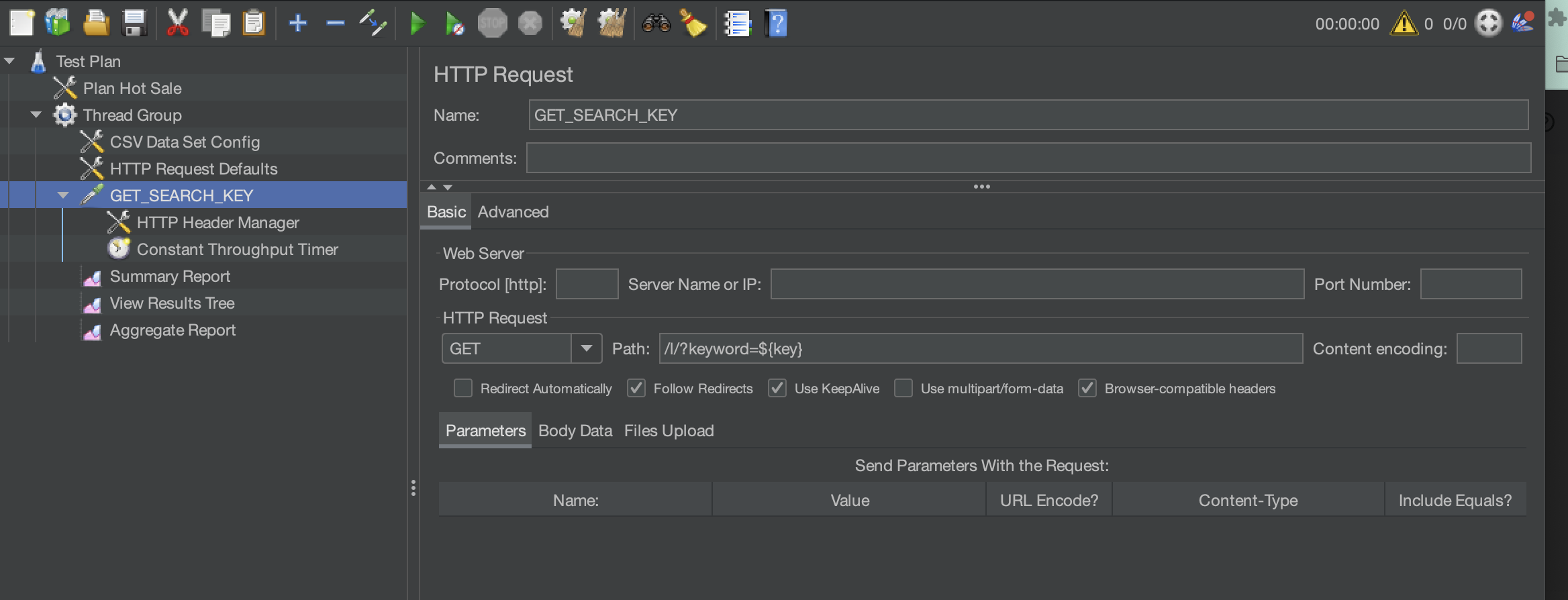
-En este ejemplo asumo que voy a ejecutar un request HTTP GET a un endpoint de búsqueda por lo que hago uso del CSV Data Set Config para tomar datos de un archivo e iterar en cada hit:



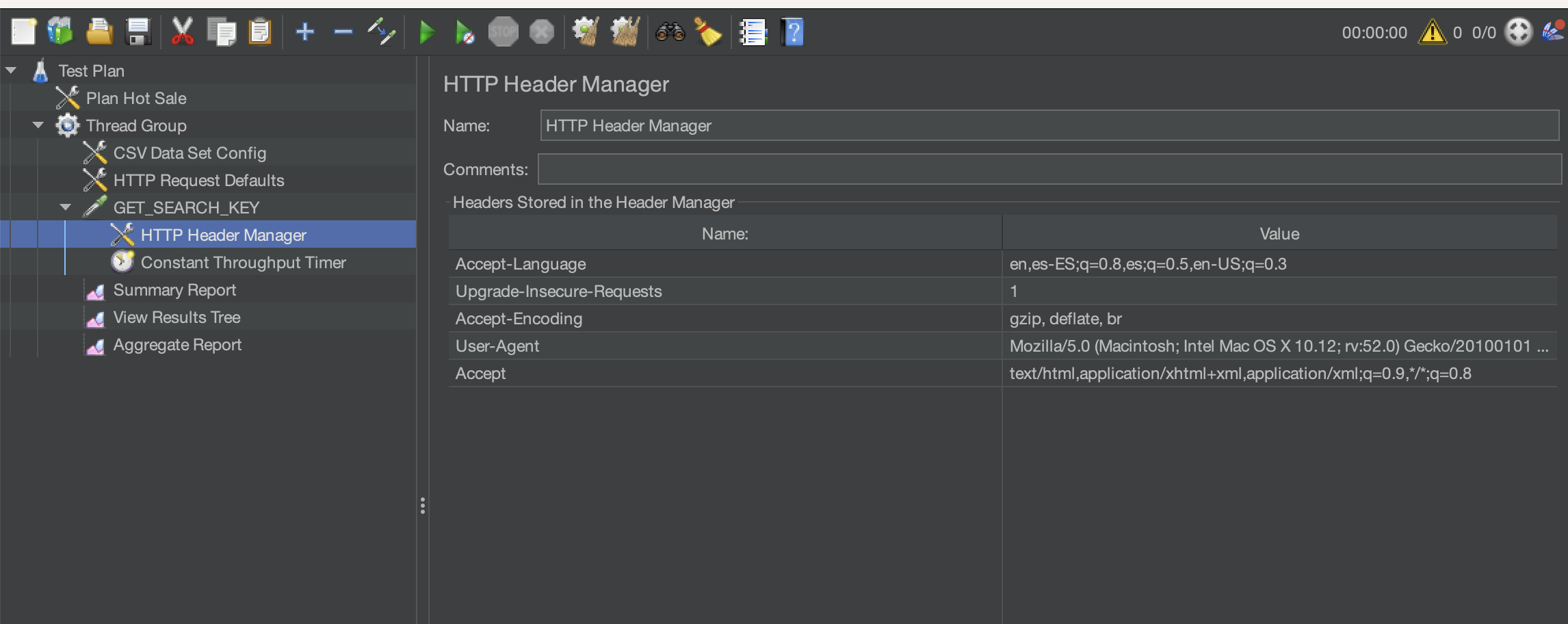
-Hago uso del HTTP Request Default para la data común entre los request y evitar duplicaciones (en este ejemplo igualmente solo tenemos un único request):



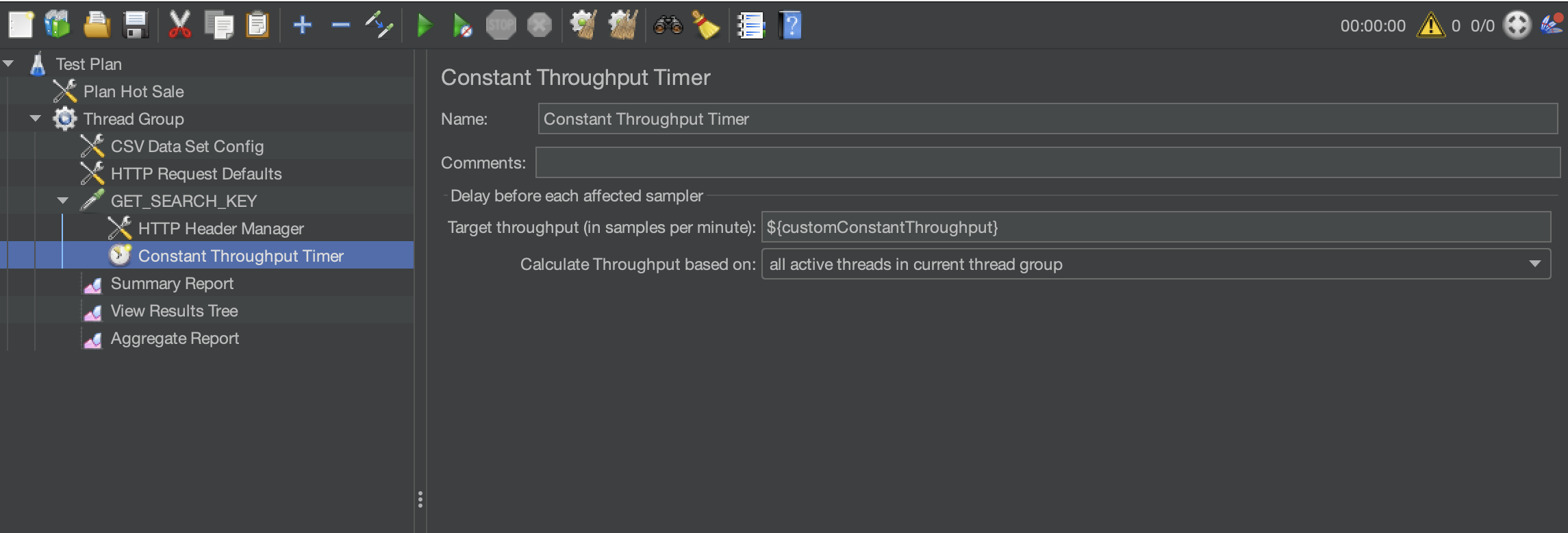
-El request en si, en donde se puede ver la referencia a la variable del csv:



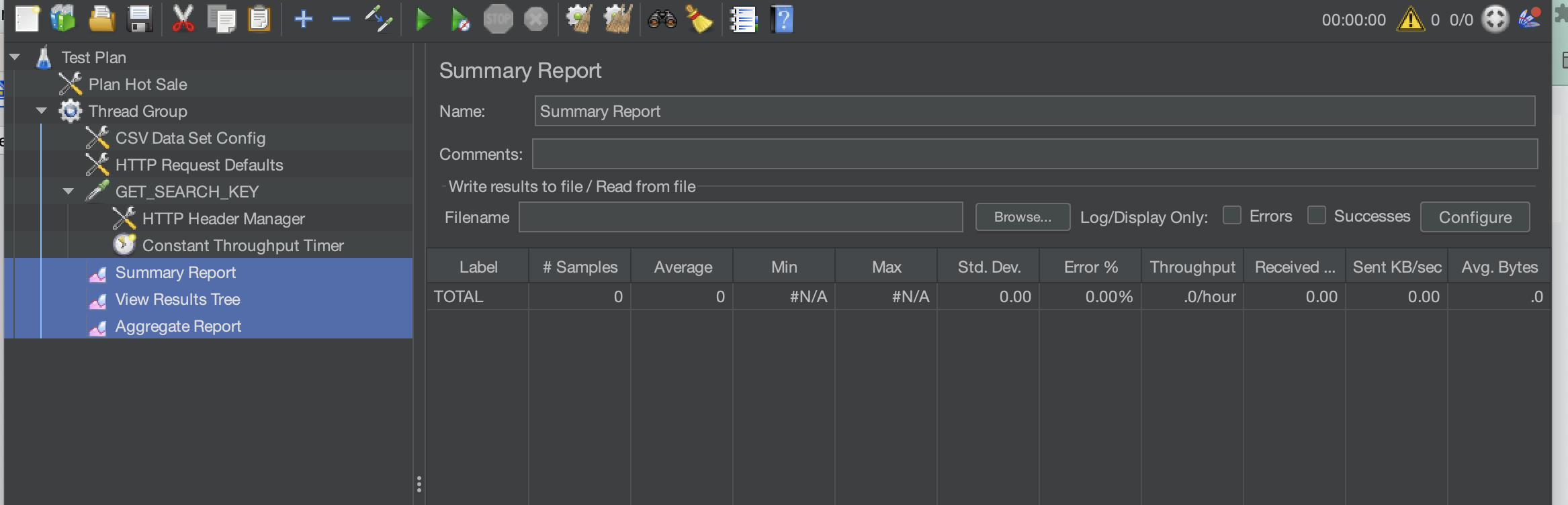
-Headers del request:



-Constant Throughput con el RPM objetivo:



-Reportes con distinto tipo de detalle, el que mas utilizaba era el aggregate report con la métrica del 90pctl:



**API Testing:**

Para probar una api intentariá contar con todas las urls necesarias en el entorno a verificar, asi como tambien ejemplos de payloads y credenciales en caso de que los endpoints requieran algun tipo de autenticación. Tambien indagaría si se cuenta con algun tipo de documentaci´øn de api, como swagger por ejemplo.

En la validación de un endopint tendría en cuenta que se responda al metodo HTTP determinado.

Tambien validaría que el formato de la respuesta sea el esperado.

Que los datos de dicha respuesta sean los esperados ya sea que previamente consulté la fuente de datos como asi podria ser una base de datos u otro request.

Como pre req entiendo por pre-requisitos a resolver antes de alguna ejecución, en ese caso podemos suponer que una api necesita un token entonces previamente debemos obtenerlos utilizando ciertas credenciales para finalmente poder ejecutar el request que necesitamos.

Para el ejemplo propuesto, manualmente en post man podria primero hacer un request para una busqueda, si puedo obtener el detalle del producto a continuación armaria imagino un POST para crear el carrito con dicho producto y si tengo algun tipo de identificaci´øn del carrito podria enviarle alguna actualización con un metodo PATCH para eliminar el producto por ejemplo.

Tal vez estos requests puedan ser parte de un script y con un job en jenkins se podria hacer la ejecución croneada, mismo a traves de algun plugin podria dispararse alguna notificación en alguna tool como slack por ejemplo.