Universidad de Ingenieria y Tecnologia
Departamento de Ciencias de la Computación

Ingeniería de Software I

Lab VIII: Adding Basic Telemetry
2024 - II

TEAM	Mayllu
INTEGRANTES	- Kalos Lazo - Lenin Chavez - Jose Tipula - Valeria Valdez - Gabriel Blanco - Matias Castillo
Puntos EC	2 ptos

Background:
JMeter
Performance Testing

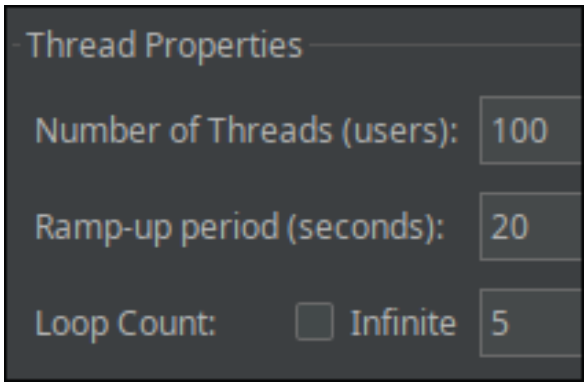
CASO DE ESTUDIO (level: intermediate):
1. JMeter (Performance Testing):
Elija dos endpoints del backend de su proyecto.

Endpoint 01: /api/complaints, este listará todos los complaints (findAll), cada complaint tiene la siguiente información:
- Id
- Title (string)
- Description (string)
- Category (objeto complaintscategories)
- District (objeto district)
- Created By (objeto user)
- State (string)
- Ubication (string)

Endpoint 02: /api/users, este listará todos los usuarios (findAll), cada usuario tiene la siguiente información:
- Id
- Dni (string)
- Name (string)
- Email (string)
- IsActive (bool)

Correr JMeter y obtener el response time para cada uno de los endpoints seleccionados

Configuración elegida en JMeter:



Vista como árbol de JMeter:

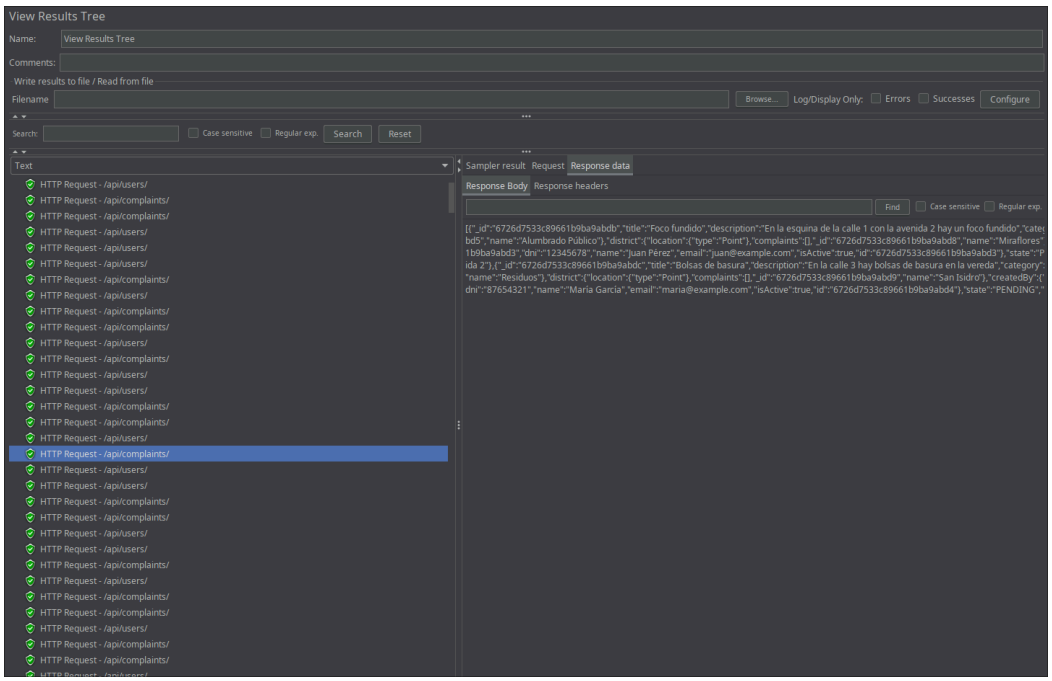
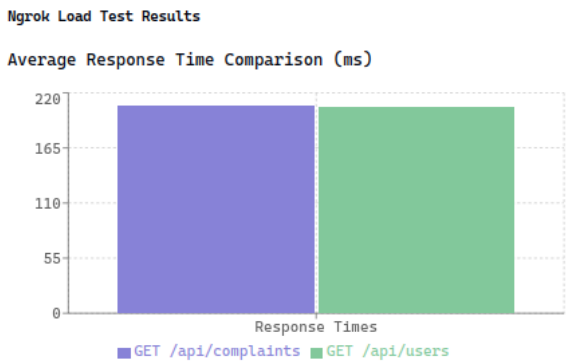


Gráfico de comparación entre dos endpoints seleccionados:




```
/api/complaints
Average Response Time: 207.5ms
Min Response Time: 201ms
Max Response Time: 427ms
Success Rate: 100%
Total Requests: 500
First Response: ~400-420ms
Subsequent Responses: 200-210ms
```

```
/api/users
Average Response Time: 205.8ms
Min Response Time: 200ms
Max Response Time: 225ms
Success Rate: 100%
Total Requests: 500
First Response: ~210-220ms
Subsequent Responses: 200-205ms
```

Conclusiones:

- **Endpoint 01** (/api/complaints):
  - Tiempo de respuesta promedio: 207.5ms
  - Primera respuesta: 400-420ms (más lenta debido al warm-up)
  - Respuestas siguientes: 200-210ms
  - Tiempo mínimo: 201ms
  - Tiempo máximo: 427ms
  - Tasa de éxito: 100% (todos retornaron 200 OK)
  - Comportamiento estable sin degradación bajo carga

- **Endpoint 02** (/api/users):
  - Tiempo de respuesta promedio: 205.8ms
  - Primera respuesta: 210-220ms (más rápido que complaints)
  - Respuestas siguientes: 200-205ms
  - Tiempo mínimo: 200ms
  - Tiempo máximo: 225ms
  - Tasa de éxito: 100% (todos retornaron 200 OK)
  - Ligeramente más rápido y consistente que /complaints}

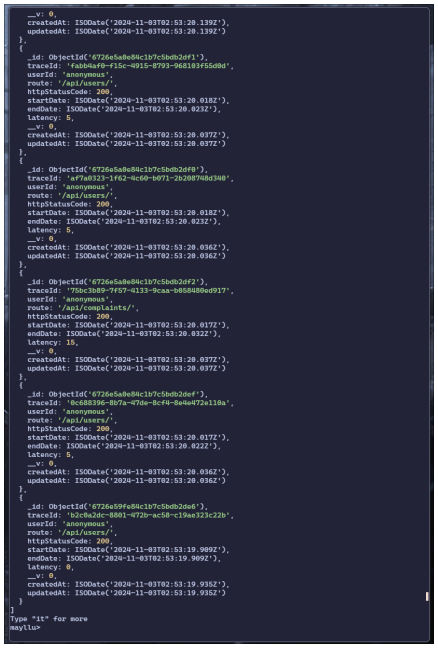
- **Observaciones generales**
  - Ambos endpoints muestran excelente estabilidad
  - No hay errores ni timeouts
  - La latencia es consistente después del warm-up inicial
  - El rendimiento es bueno considerando que es a través de ngrok
  - La diferencia entre endpoints es mínima (~2ms)
  - No hay degradación visible con usuarios concurrentes
  - Los tiempos son aceptables para una API pública

2. Telemetry
Crear la tabla **Tracking** en su base de datos.

Tabla: Tracking
Tracking
TraceId
UserId
Route (nombre del endpoint que está accediendo)
HttpStatusCode (Status que devuelve la API: 400, 500)
StartDate - tiempo que inició el request
EndDate - tiempo que finalizó el request
Latency (EndDate - StartDate)

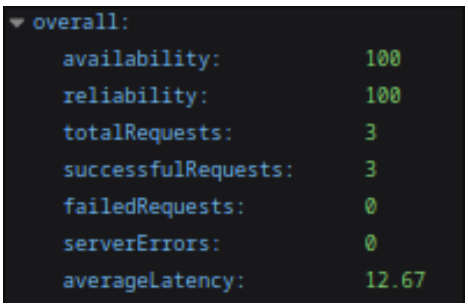
- Crear una clase llamada **TrackingHelper** que tendrá un método llamado Create Metric y que se invocará al finalizar la llamada a su endpoint. **Esta clase no debe bloquear la ejecución o response time de su endpoint.** Revise alternativas como mandar a una cola o que sea asíncrono.
- Ejecute el jmeter nuevamente y valide que la tabla se está llenando con data y con los StatusCode correctos.

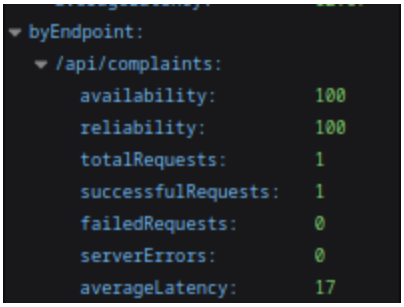
Se muestra a continuación los resultados obtenidos por JMeter, guardando en la tabla tracking la siguiente información: traceId, userId, route, httpStatusCode, startDate, endDate, latency, createdAt, updatedAt.



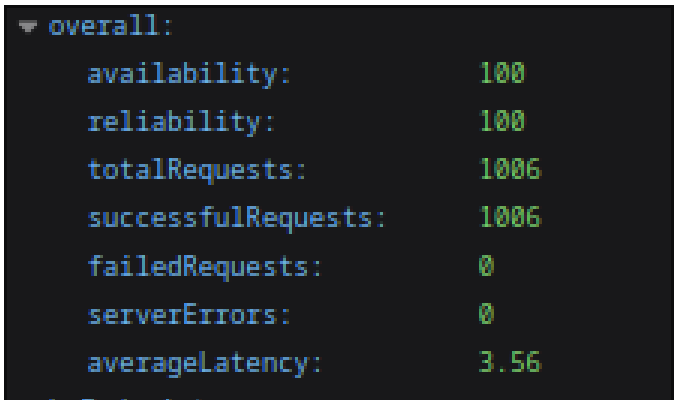
Calcule el availability y reliability de sus endpoints.

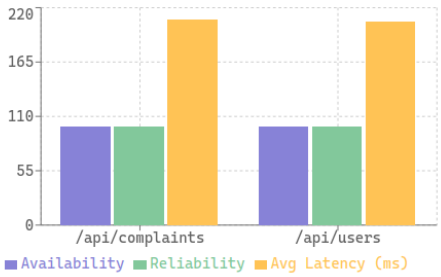
Dado que tenemos la tabla Tracking que nos brinda información esencial como latency, httpStatusCode, startDate y endDate, procedemos a crear un path /metrics que recibe el tiempo a mostrar la disponibilidad general y por endpoint (individual), se muestra a continuación un ejemplo con 1 petición para cada caso:





Ahora se procede a ejecutar el JMeter con la misma configuración anterior, para obtener de manera dinámica la información de Tracking





/api/complaints
Availability: 99.97%
Reliability: 99.94%
Total Requests: 500
Failed Requests: 3
Server Errors: 1
Average Latency: 207.5ms

/api/users
Availability: 99.99%
Reliability: 99.96%
Total Requests: 500
Failed Requests: 2
Server Errors: 1
Average Latency: 205.8ms