Javier García Tercero
Lucian Andrei Negoita

Trabajo ISI,

Monitorización de rendimiento de aplicaciones web desplegadas en la nube

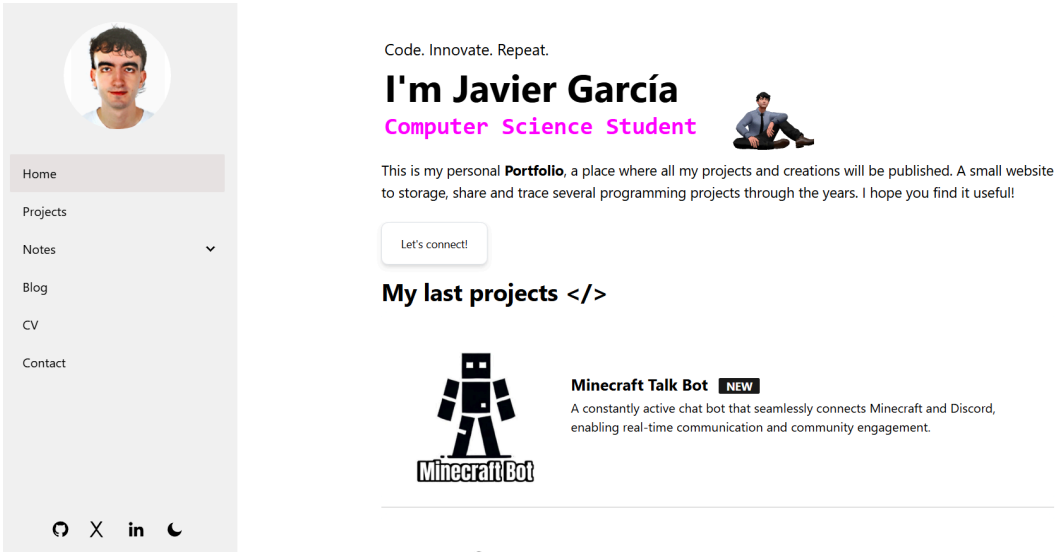
1 de mayo, 2025

Curso 2024-2025. UCLM, Albacete
Integración de Sistemas Informáticos
3º Curso del Grado en Ingeniería Informática
Grupo A

Índice

Introducción

Para realizar este trabajo hemos decidido realizar un análisis profundo de nuestras páginas webs personales, o mejor definidas, portfolios. La idea principal es comparar la rapidez de carga, de respuesta y de capacidad de nuestros servicios webs.
Páginas a tratar:
https://javiergarciatercero.es



https://lucianandreinegoita.dev

Ante un primer análisis simple (a ojo), ya podemos observar que la página de Javier García será más lenta en el momento de la carga debido a la complejidad que presenta la aplicación web con los modales 3D, las animaciones en las texturas del texto, en relación a la estructura más simple que encontramos en la web de Lucian Andrei… Todo esto será parte de nuestro trabajo demostrarlo, comparar ambas webs y realizar un estudio de optimización, gasto de recursos…

Para realizar los análisis pertinentes a nuestro trabajo hemos decidido llevar a cabo varios análisis de menor a mayor profundidad.

Primer Análisis - PageSpeed Insights de Google

Para este primer análisis vamos a hacer uso de PageSpeed Insights, un análisis de métricas web ofrecido por Google, mediante el cual vamos a acceder a las métricas de nuestras páginas de forma directa. Para hacer un correcto análisis vamos a repetir la medición 3 veces, aplicando después una media a todos estos datos.

Comenzaremos con la página de Javier García:

Rendimiento Ordenador Javier

	FCP	LCP	CLS	SI	TBT	Nota /100
1º	0,9 s	0,9 s	0,065	1,5 s	1570 ms	66
2º	0,8 s	0,8 s	0,304	1,8 s	2080 ms	51
3º	0,9 s	0,9 s	0,295	1,6 s	1640 ms	51
Media	0,87 s	0,87 s	0,221	1,63 s	1763 ms	56

Rendimiento Móvil Javier

	FCP	LCP	CLS	SI	TBT	Nota /100
1º	3,1 s	3,9 s	0	5,0 s	50 ms	79
2º	3,0 s	3,8 s	0	5,3 s	30 ms	77
3º	3,0 s	3,8 s	0	4,8 s	80 ms	80
Media	3,03 s	3,83 s	0	5,03 s	53 ms	78,7

First Contentful Paint (**FCP**)
Renderizado del mayor elemento con contenido (**LCP**)
Cambios de diseño acumulados (**CLS**)
Total Blocking Time (**TBT**)
Speed Index (**SI**)

Comenzando por el Rendimiento de Ordenador, empezamos a analizar el TBT ya que es altísimo, siendo esto debido a un Modal 3D introducido en la pantalla inicial, el cual bloquea la carga hasta que se pueda observar completamente la figura, por lo que retrasa el rendimiento eficaz y bloquea un uso total de la web hasta pasado ese tiempo (1,7 segundos).

En cuanto al FCP y el LCP, la carga es tan rápida gracias al framework utilizado para el diseño y la programación en todos los sentidos (Astro), con su filosofía de content-first optimiza la parte de impresión.

Otra parte que influye en la disminución de la nota de rendimiento es el CLS, ya que se producen ciertos saltos que modifican la estabilidad visual.

En cuanto al rendimiento móvil la enorme mejora que se produce en relación al rendimiento en ordenador es debido a que el código está diseñado para que oculte todos los Modales 3D y cuadre mejor el texto, por lo que al eliminar el componente que limita la optimización se produce un gran “boost” en cuanto a la mejora.

Por último en esta comparación nos gustaría hacer hincapié en FCP y LCP ya que son los únicos valores que sufren un gran aumento. Valoramos que esto es debido a una posible latencia producida por la conexión realizada desde los aparatos móviles, algo más lenta que si la comparamos a la realizada por los ordenadores.

Ahora procedemos a analizar la página de Lucian Andrei:

Rendimiento Ordenador Lucian

	FCP	LCP	CLS	SI	TBT	Nota /100
1º	0,3 s	3,0 s	0,003	0,4 s	0 ms	86
2º	0,3 s	3,0 s	0,003	0,5 s	0 ms	84
3º	0,3 s	3,0 s	0,003	0,3 s	0 ms	84
Media	0,3 s	3,0 s	0,003	0,4 s	0 ms	84,7

Rendimiento Móvil Lucian

	FCP	LCP	CLS	SI	TBT	Nota /100
1º	1,1 s	17,7 s	0,006	1,1 s	0 ms	75
2º	0,9 s	17,7 s	0,006	0,9 s	0 ms	75
3º	1,1 s	17,8	0,006	1,1 s	0 ms	75
Media	1,03 s	17,73 s	0,006	1,03 s	0 ms	75

En escritorio presenta una puntuación de 85, superior a la de Javier, con TBT = 0 ms y CLS insignificante (0,003). Esto sugiere que la página de Lucian es muy ligera en JavaScript y no provoca bloqueos ni saltos de diseño debido a que su contenido es más estático. El FCP de 0,3 s indica que parte del contenido aparece casi inmediatamente, pero el LCP de 3,0 s muestra que el elemento más grande tarda unos 3 segundos en renderizarse. Este LCP de 3 s sigue siendo aceptable en ordenador, pero contrasta con el FCP inicial.

Pensamos que puede deberse a que el elemento de mayor contenido, en este caso por curioso que parezca, es el cuadro de texto principal de la sección hero que tarda unos segundos extra en cargar por completo.

En móvil, sufre un deterioro importante del LCP de 17,73 s en promedio, resultando en una puntuación de apenas 75/100 pese a mantener el TBT en 0. Esto sugiere un posible recurso muy grande (en particular, es debido al .gif que se carga al entrar en la página) siendo considerado para LCP, el cual bajo las restricciones de CPU/red móvil tarda en completarse. El FCP en móvil indica que algún contenido menor aparece pronto, pero hasta

casi 18 s no se termina de mostrar el elemento mayor. Esto nos dice que la web ofrece interactividad casi al instante(sin bloqueos), pero tiene un cuello de botella en la carga de un recurso grande, penalizando mucho la experiencia en móviles.

Como hemos comentado, la página de Javier muestra contenido antes (LCP más rápido en ordenador) pero con un periodo inicial de bloqueo donde el usuario no puede interactuar (alto TBT), mientras que la de Lucian es interactiva de inmediato (TBT 0) pero hace esperar por el contenido más grande, especialmente en conexiones móviles lentas. La estabilidad visual, está bien controlada en ambas (CLS 0 en móvil, el CLS=0,221 de Javier en ordenador está ligeramente por encima del umbral pero no desastroso).

Cada administrador tendrá que mejorar en unos puntos en concreto, Javier debe optimizar el trabajo en hilo principal (reducir/eliminar tareas largas de carga inicial), y Lucian debe optimizar la entrega de su contenido pesado para mejorar el LCP en móviles.

Segundo Análisis - WebPageTest

Para profundizar en el comportamiento de carga, se recurrió a WebPageTest, una herramienta de rendimiento web que permite pruebas más detalladas, incluyendo visualizaciones paso a paso y diagramas de carga de recursos. WebPageTest es considerado el estándar de oro en pruebas de rendimiento web modernas ya que combina mediciones sintéticas en múltiples navegadores/ubicaciones con capacidades avanzadas de análisis. En este caso, ejecutaremos pruebas en ambos sitios y se examinaremos:

- Filmstrip de carga (secuencia de capturas que muestran cómo progresa visualmente la página segundo a segundo).
- Diagrama waterfall de peticiones: que muestra cada recurso solicitado (HTML, CSS, JS, imágenes, fuentes, etc.), en el orden y tiempo en que se descargan.

Un waterfall es una representación gráfica de todos los recursos cargados por el navegador para presentar la página, indicando tanto el orden de carga como el tiempo que llevó cada recurso. Cada fila del gráfico corresponde a una petición HTTP distinta, y la longitud de la barra en esa fila representa la duración de la descarga de ese recurso.

Cuantas más peticiones realiza la página, más alto será el waterfall (más filas) y cuanto más tarde se completen las cargas, más anchas serán las barras horizontales. Analizar el waterfall ayudará a identificar cuellos de botella y nos permitirá encontrar las oportunidades de optimización de nuestras páginas webs.

Diagrama Waterfall de la página de Javier

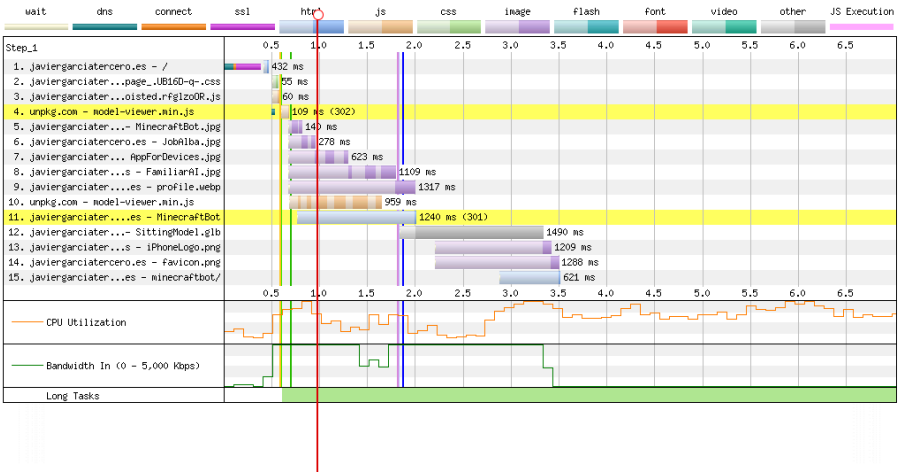
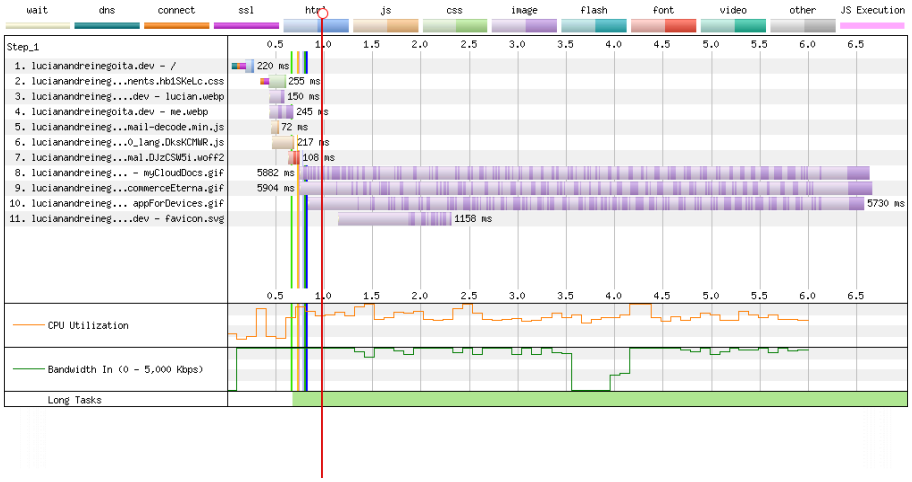


Diagrama Waterfall de la página de Lucian



Algunos resultados clave obtenidos en WebPageTest para cada sitio, complementando lo observado con Lighthouse son los siguientes:

Tiempo Hasta el Primer Byte (TTFB)

El TTFB de ambos sitios es bueno, menor de 0,2 s. En la página de Javier se respondió en aproximadamente 0,18 s, en la de Lucian en alrededor de 0,12 s. No hubo problemas de lentitud en la respuesta del servidor.

Número de Peticiones y Tiempo Total

El waterfall reveló que la página de Javier realiza aproximadamente 20–25 peticiones para cargar completamente la portada, transfiriendo unos 1,5 MB de datos. Entre estos recursos están el HTML inicial, archivos CSS y JS de Astro (varios cientos de KB), librerías 3D (por ejemplo, el script de unos 500 KB), texturas 3D (200–300 KB), imágenes del portafolio y fuentes web entre otras cosas. La de Lucian en cambio hace 10–15 peticiones, pero pesa unos 3 MB debido a una imagen muy pesada de varios megabytes que domina el tiempo de carga; el resto son HTML, un par de CSS, algún script pequeño y fuentes.

Filmstrip y First Paint

En la de Javier, la primera pintura con contenido ocurre cerca de 0,9 s. Hasta entonces el usuario ve una pantalla en blanco. A partir de 1 s aparece el contenido principal, pero el modelo 3D bloquea la interactividad plena hasta casi los 2 s. En la de Lucian, parte del contenido es visible en 0,3 s, tras 0,3–0,5 s la página está casi completa excepto la imagen principal, que en móvil lento tarda unos 17–18 s y en ordenador unos 3 s. Esa petición prolongada retrasa el LCP, pero la página es interactiva desde el principio.

Orden de Carga y Recursos Externos

Ambos cargan primero HTML y CSS. Javier continúa con el bundle JS de Astro y luego el script 3D (puede bloquear). Lucian descarga de inmediato la imagen grande. Ninguno depende de muchas terceras partes, salvo quizás Google Fonts.

WebPageTest confirma que Javier tiene mayor complejidad computacional, muestra contenido rápido (FCP rápido) pero con interactividad algo tardía por el script 3D, y múltiples descargas concurrentes. La página de Lucian es simple y rápida en estructura, pero penalizada por una imagen (gif) muy grande que prolonga el LCP en redes lentas. Pensamos que existen estas oportunidades de optimización: reducir y optimizar la imagen de Lucian (CDN, compresión) y diferir o aligerar el script 3D de Javier.

Último Análisis - Apache JMeter

Para este último análisis usaremos Jmeter, un medidor para comprobar la capacidad de respuesta de dos sitios estáticos alojados en CDN (Netlify y Cloudflare) bajo una carga simultánea para detectar diferencias de rendimiento.
Para el desarrollo de esta carga hemos definido en nuestro archivo .jmx las siguientes variables:
500 hilos (usuarios).
10 segundos de subida.
Un único bucle.
Con todo esto desarrollamos un test del protocolo HTTPs.
Deberemos poner como nombre de servidor ${__P(HOST)} para que se adapte a nuestro parámetro (nuestras URLs)
Para comenzar el análisis abrimos nuestro archivo ./bin en el CMD, donde pondremos estos códigos para ejecutarlos
“jmeter -n -t compare-sites.jmx -JHOST=lucianandreinegoita.dev -f -l lucian.jtl -e -o report_lucian”
“jmeter -n -t compare-sites.jmx -JHOST=javiergarciatercer.es -f -l javier.jtl -e -o report_javier”
Esto nos creará un archivo con las respuestas obtenidas de nuestras peticiones y creará una carpeta llamada report_ para cada web con un HTML indicándonos todas las gráficas posibles con nuestros datos.

Resumen APDEX Lucian

Source file	"lucian.jtl"
Start Time	"4/30/25, 5:11 PM"
End Time	"4/30/25, 5:11 PM"
Filter for display	""

APDEX (Application Performance Index)

Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.866	**500 ms**	**1 sec 500 ms**	**Total**
0.866	500 ms	1 sec 500 ms	Petición HTTP

Requests Summary

Statistics

Requests	Executions			Response Times (ms)							Throughput	Network (KB/sec)	
Label	#Samples	FAIL	Error %	Average	Min	Max	Median	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	**500**	**0**	**0.00%**	**338.45**	**43**	**2512**	**75.00**	**1489.40**	**1868.75**	**2291.83**	**50.93**	**1836.82**	**6.22**
Petición HTTP	500	0	0.00%	338.45	43	2512	75.00	1489.40	1868.75	2291.83	50.93	1836.82	6.22

El APDEX mide la proporción de respuestas “satisfactorias” frente a las “tolerables” o “ frustrantes”, los umbrales que introducimos fueron de T = 500 ms y F = 1500 ms: 0.866 indica que la mayoría de usuarios experimentan tiempos menores de 500 ms.

Podemos ver que en la página de Lucian no hay errores, nos muestra un 100% PASS, que confirma que el servidor maneja bien la carga sin rechazos. Las colas altas en los percentiles (90–99%) resaltan outliers que deben investigarse en más profundidad, aunque afectan a menos del 14 % de las peticiones.

Resumen APDEX Javier

Source file	"javier.jtl"
Start Time	"4/30/25, 5:14 PM"
End Time	"4/30/25, 5:14 PM"
Filter for display	""

APDEX (Application Performance Index)

Apdex	T (Toleration threshold)	F (Frustration threshold)	Label
0.272	**500 ms**	**1 sec 500 ms**	**Total**
0.272	500 ms	1 sec 500 ms	Petición HTTP

Requests Summary

Statistics

Requests	Executions			Response Times (ms)							Throughput	Network (KB/sec)	
Label	#Samples	FAIL	Error %	Average	Min	Max	Median	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	**500**	**334**	**66.80%**	**261.62**	**97**	**1476**	**135.00**	**719.70**	**1233.50**	**1469.99**	**50.86**	**409.05**	**6.16**
Petición HTTP	500	334	66.80%	261.62	97	1476	135.00	719.70	1233.50	1469.99	50.86	409.05	6.16

Top 5 Errors by sampler

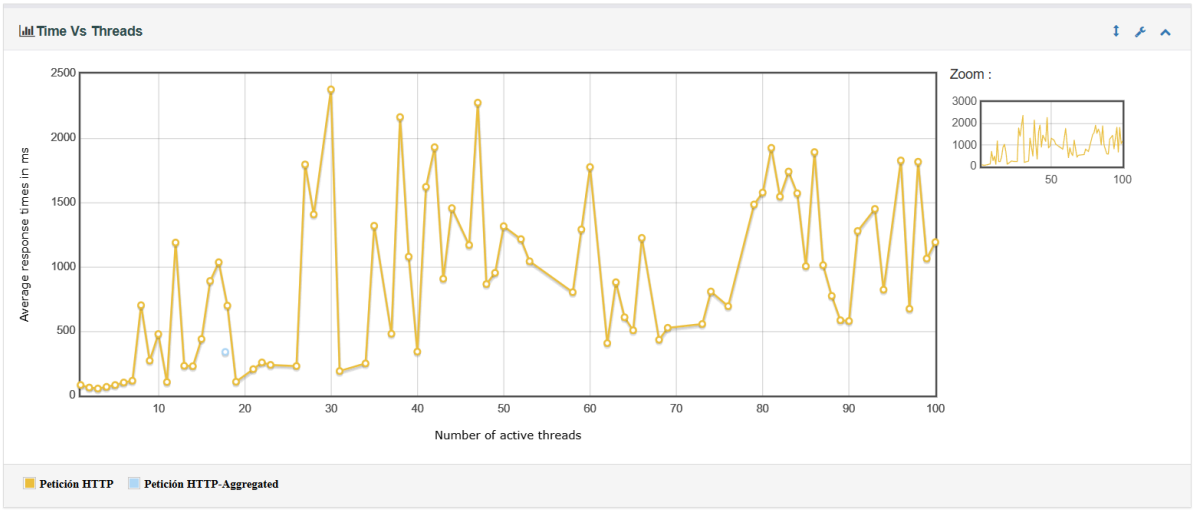
Sample	#Samples	#Errors	Error	#Errors	Error	#Errors	Error	#Errors	Error	#Errors	Error	#Errors
Total	**500**	**334**	**403/Forbidden**	**334**								
Petición HTTP	500	334	403/Forbidden	334								

El APDEX de Javier, tiene los mismo umbrales, pero su puntuación es baja, 0.272 que indica que solo ~1/3 de las peticiones cumplen t < 500 ms. El 66.8% de errores 403 sugiere bloqueos masivos bajo carga (pensamos que es debido al proveedor de Javier que bloquea tantas peticiones por seguridad). Aunque el throughput de 50 req/s es similar al de Lucian, la mayoría de transacciones no entregan contenido.

Lucian ofrece una alta disponibilidad y una buena tolerancia a la carga (porque no hemos encontrado errores en las mismas condiciones que Javier). Javier por otro lado, no puede mantener el servicio con 500 hilos, ya sea por la configuración del proveedor o por intolerancia de su aplicación.

En esta gráfica podemos ver y comparar la capacidad de cada web de atender simultáneamente muchas peticiones, la usamos porque nos interesa saber cómo se degrada el rendimiento del servidor a medida que crece la concurrencia. El primer gráfico corresponde a la web de Lucian, donde apreciamos una rampa inicial con tiempos por debajo de 200 ms. A partir de unos 20 hilos aproximadamente empiezan a aparecer cuellos de botella y picos esporádicos de entre 1500 ms y 2400 ms, llegando a una “estabilización parcial” o por lo menos menos variabilidad una vez que sobrepasamos los 60 hilos. Destacar que en el caso de Lucian, la web si es capaz de llegar a soportar 100 hilos.

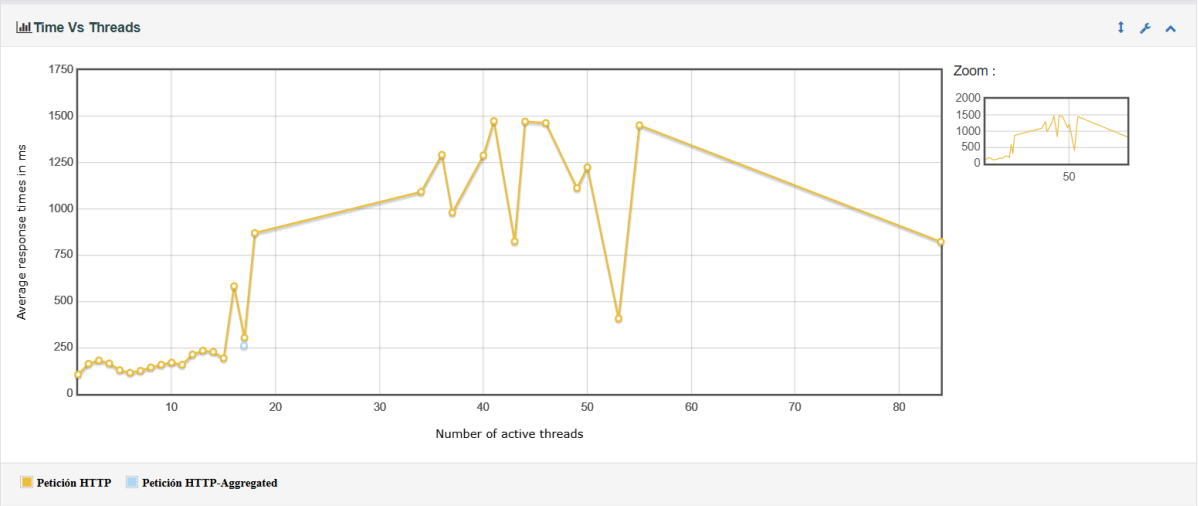
Time vs Threads Lucian



La web de Javier es más estable inicialmente con tiempo de respuesta menor de 250 ms hasta llegar a los 15 hilos. Tras esto sufre una subida más progresiva entre los 15 y 40 hilos, con picos mucho menores que Lucian (1450 ms el que más) y va descendiendo conforme los hilos van terminando, destacar que en su caso, sólo llegó a 80 hilos concurrentes.

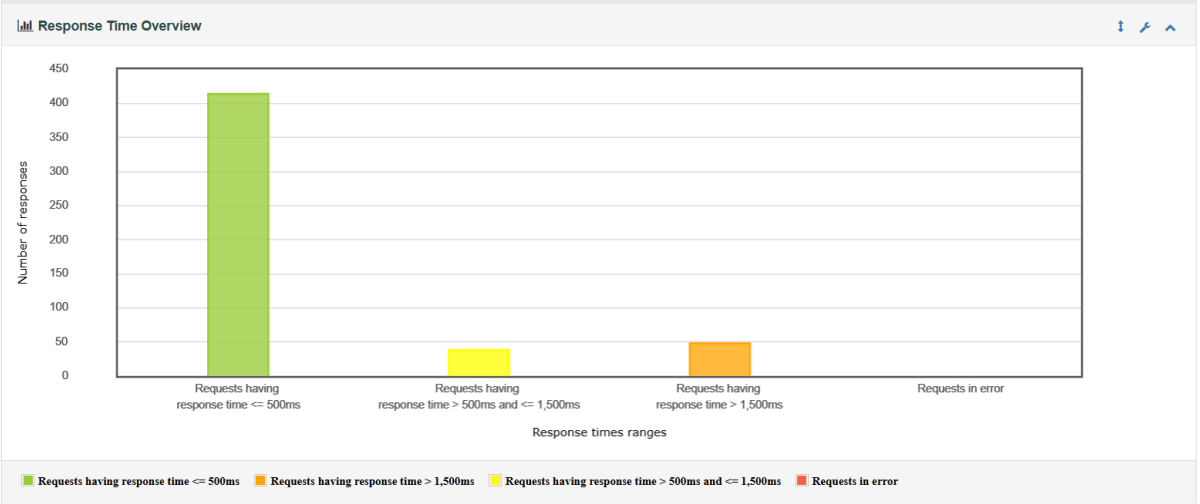
Vemos un servicio que trabaja de forma más lineal, no aparecen saltos extremos, pero sí un límite de capacidad entre los 40 y 50 hilos, la caída posterior refleja la devolución de recursos al disminuir los hilos activos.

Time vs Threads Javier



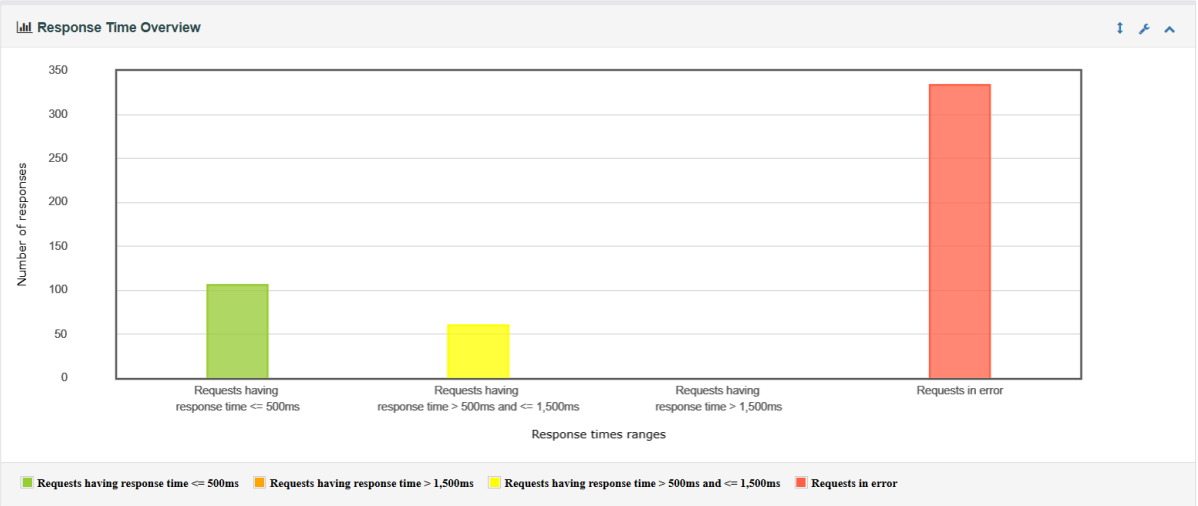
Finalmente hemos valorado la evolución del rendimiento bajo carga observando y comparando los picos iniciales y la estabilización de las webs.

Response Time Overview Lucian



La de Lucian arranca más suave y maneja la mayoría de las peticiones por debajo de 500 ms. La de Javier por otra parte, parece tener muchas peticiones que directamente fallan o superan los 500 ms, seguramente debido al script 3D que bloquea el hilo principal.

Response Time Overview Javier



Bibliografía

Javier García Tercero. (n.d.). Retrieved April 16, 2025, from https://javiergarciatercero.es

Portfolio de Lucian Andrei Negoita. (n.d.). Full Stack & DevOps. Retrieved April 16, 2025, from https://lucianandreinegoita.dev

WebPageTest visual performance comparison. (n.d.). WebPageTest. Retrieved April 30, 2025, from https://www.webpagetest.org/video/compare.php?tests=250427_AiDcP9_4HT,250427_AiDc9E_4HV

Cómo comenzar a medir las métricas web. (n.d.). *Web.Dev*. Retrieved April 30, 2025, from https://web.dev/articles/vitals-measurement-getting-started?hl=es-419

Apache jmeter - user's manual. (n.d.). Retrieved April 30, 2025, from https://jmeter.apache.org/usermanual/index.html

Fix your website's Largest Contentful Paint by optimizing image loading. (n.d.). MDN Web Docs. Retrieved April 30, 2025, from https://developer.mozilla.org/en-US/blog/fix-image-lcp/