UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN

Integración de Dapper para generar reportes en el Sistema de Gestión de Investigación de la Universidad Nacional de Chimborazo

Trabajo de Titulación para optar al título de Ingeniero en Tecnologías de la Información

Autor:
Becerra Hidalgo, Valeria Alexandra

Tutor:

Ph.D. Molina Granja, Fernando Tiverio

Riobamba, Ecuador. 2025

DECLARATORIA DE AUTORÍA

Yo, Valeria Alexandra Becerra Hidalgo, con cédula de ciudadanía 0605729433, autora del trabajo de investigación titulado: Integración de Dapper para generar reportes en el Sistema de Gestión de Investigación de la Universidad Nacional de Chimborazo, certifico que la producción, ideas, opiniones, criterios, contenidos y conclusiones expuestas son de mí exclusiva responsabilidad.

Asimismo, cedo a la Universidad Nacional de Chimborazo, en forma no exclusiva, los derechos para su uso, comunicación pública, distribución, divulgación y/o reproducción total o parcial, por medio físico o digital; en esta cesión se entiende que el cesionario no podrá obtener beneficios económicos. La posible reclamación de terceros respecto de los derechos de autor (a) de la obra referida, será de mi entera responsabilidad; librando a la Universidad Nacional de Chimborazo de posibles obligaciones.

En Riobamba, a los 27 días del mes de mayo del 2025.

Valeria Alexandra Becerra Hidalgo

C.I: 0605729433

ACTA FAVORABLE - INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN

En la Ciudad de Riobamba, a los 23 días del mes de abril de 2025, luego de haber revisado el Informe Final del Trabajo de Investigación presentado por el estudiante **Valeria Alexandra Becerra Hidalgo** con CC: **0605729433**, de la carrera **INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN** y dando cumplimiento a los criterios metodológicos exigidos, se emite el **ACTA FAVORABLE DEL INFORME FINAL DEL TRABAJO DE INVESTIGACIÓN** titulado **"INTEGRACIÓN DE DAPPER PARA GENERAR REPORTES EN EL SISTEMA DE GESTIÓN DE INVESTIGACIÓN DE LA UNIVERSIDAD NACIONAL DE CHIMBORAZO"**, por lo tanto se autoriza la presentación del mismo para los trámites pertinentes.

PhD. Fernando Molina Granja
TUTOR

CERTIFICADO DE LOS MIEMBROS DEL TRIBUNAL

Quienes suscribimos, catedráticos designados Miembros del Tribunal de Grado para la evaluación del trabajo de investigación: Integración de Dapper para generar reportes en el Sistema de Gestión de Investigación de la Universidad Nacional de Chimborazo, presentado por Valeria Alexandra Becerra Hidalgo, con cédula de identidad número 0605729433, bajo la tutoría de Ph.D. Fernando Tiverio Molina Granja; certificamos que recomendamos la APROBACIÓN de este con fines de titulación. Previamente se ha evaluado el trabajo de investigación y escuchada la sustentación por parte de su autor; no teniendo más nada que observar.

De conformidad a la normativa aplicable firmamos, en Riobamba a los 27 días del mes de mayo de 2025.

Ana Congacha, Mgs.
PRESIDENTE DEL TRIBUNAL DE GRADO

Lady Espinoza, Mgs.
MIEMBRO DEL TRIBUNAL DE GRADO

Danny Velasco, Mgs
MIEMBRO DEL TRIBUNAL DE GRADO

CERTIFICACIÓN

Que, **BECERRA HIDALGO VALERIA ALEXANDRA** con CC**: 0605729433**, estudiante de la Carrera de **INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN**, Facultad de **INGENIERÍA;** ha trabajado bajo mi tutoría el trabajo de investigación titulado **"INTEGRACIÓN DE DAPPER PARA GENERAR REPORTES EN EL SISTEMA DE GESTIÓN DE INVESTIGACIÓN DE LA UNIVERSIDAD NACIONAL DE CHIMBORAZO"**, cumple con el **6** %, de acuerdo al reporte del sistema Anti plagio COMPILATIO, porcentaje aceptado de acuerdo a la reglamentación institucional, por consiguiente autorizo continuar con el proceso.

Riobamba, 16 de mayo de 2025

FERNANDO TIVERIO MOLINA GRANJA

PhD. Fernando Molina Granja, PD
TUTOR

DEDICATORIA

Dedico este trabajo a Dios, quien me ha dado la fuerza, la sabiduría y la perseverancia que necesito para terminar esta etapa. A mis padres **Gustavo Becerra** y **Nelly Hidalgo**, por su amor incondicional, sus sacrificios y constante apoyo, que han sido pilares fundamentales en cada paso de mi formación.

A mis hermanos **Nancy**, **Laura**, **Gladys**, **Gustavo** y **Anita**, por sus palabras de aliento y comprensión en los momentos más difíciles de mi vida; especialmente a mi hermana **Laura**, quien ha estado a mi lado desde el inicio de esta carrera, brindándome su apoyo incondicional y constante motivación. Ella ha sido un ejemplo para mí y admiro profundamente la gran profesional que se ha convertido.

AGRADECIMIENTO

Agradezco sinceramente a Alex Asitimbay, quien me guio con su conocimiento y cuyo apoyo incondicional hizo posible este logro. Su experiencia y disposición fueron fundamentales en este proceso.

También quiero expresar mi más sincero agradecimiento a Alex Buñay, por ofrecerme su amistad y por creer en mí incluso cuando yo no lo hacía. Su constante motivación me inspiró a confiar en mi capacidad para crecer como desarrolladora. Su profesionalismo, respeto y calidad humana han sido un ejemplo que valoro profundamente.

De manera especial, deseo extender mi más sincero agradecimiento a la Dra. Magda Cejas, cuya atención y afecto fue un gran soporte emocional durante mi estadía en la Sala de Investigación. Su calidez y constante apoyo, al considerarme parte del equipo de Investigación, ha dejado una huella imborrable en mi corazón.

Finalmente, agradezco a mi tutor por su guía constante en la elaboración de los documentos, sus sabias enseñanzas y la paciencia con la que me acompañó durante esta etapa.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El objetivo de esta investigación fue desarrollar un módulo de reportes para el Sistema de Gestión de Investigación de la Universidad Nacional de Chimborazo, evaluando su eficiencia luego de la integración de los servicios web Dapper y API REST, utilizando los indicadores de la norma ISO/IEC 25010. Se adoptó una metodología cuantitativa, recolectando datos mediante Apache JMeter sobre comportamiento temporal, utilización de recursos y capacidad del sistema.

La metodología de estudio fue no experimental, examinando las variables en su ambiente natural, sin interferencias externas. Los hallazgos indicaron que el sistema compensó el 100% de las exigencias de comportamiento temporal, con periodos de respuesta de hasta 15 segundos. Respecto a los recursos, se logró un uso medio de CPU del 31.25%, RAM del 63% al 70%, y un uso de disco del 6.75%, lo que permitió un cumplimiento del 91.6%. En términos de capacidad, el sistema registró 0.83 peticiones por segundo, satisfaciendo un 83% de las necesidades.

Para sintetizar, la incorporación de Dapper perfeccionó el desempeño del sistema, respetando los criterios de eficiencia fijados por la norma ISO/IEC 25010. El sistema demostró un adecuado balance entre rapidez y eficacia, sin fallos en los ensayos, corroborando que Dapper es una opción eficaz para la creación de informes en contextos que demandan un alto rendimiento.

Palabras claves: Dapper, eficiencia, norma ISO/IEC 25010, rendimiento, reportes.

ABSTRACT

This research aimed to develop a reporting module for the Research Management System of the Universidad Nacional de Chimborazo and evaluate its efficiency after integrating Dapper web services and REST APIs using the ISO/IEC 25010 standard indicators. A quantitative methodology was necessary, and data on temporal behavior, resource utilization, and system capacity were collected using Apache JMeter. The study methodology was non-experimental, examining the variables in their natural environment without external interference. The findings indicated that the system compensated 100% of the temporal behavior demands, with up to 15 seconds response times. Regarding resources, achieving an average CPU utilization of 31.25%, RAM utilization of 63% to 70%, and disk utilization of 6.75% allowed for 91.6% compliance. Regarding capacity, the system recorded 0.83 requests per second, meeting 83% of the requirements. To summarize, adding Dapper improved the system's performance, meeting the efficiency criteria set by ISO/IEC 25010. The system demonstrated an appropriate balance between speed and efficiency, with no glitches in testing, confirming that Dapper is an effective option for reporting in high-performance environments.

Keywords: Dapper, efficiency, ISO/IEC 25010, performance, reporting.

Reviewed by:
Mgs. Jessica María Guaranga Lema
ENGLISH PROFESSOR
C.C. 0606012607

CAPÍTULO I. INTRODUCCIÓN

La Universidad Nacional de Chimborazo, se identifica por ser una institución desafiante, que motiva a la investigación científica por medio de la Dirección de Investigación, quien es la responsable de administrar y dar seguimiento a los proyectos de investigación. Con esta perspectiva, se hace evidente la importancia de abordar tecnologías emergentes, tal como la "Integración de Dapper para producir informes en el sistema de administración de investigación de la Universidad Nacional de Chimborazo. El objetivo de este estudio es elaborar informes que faciliten la elección de datos pertinentes de los proyectos. Esta solución facilita la mejora del acceso a la información, fomentando una toma de decisiones más fundamentada y eficaz dentro de la universidad.

En este contexto, se eligió el desarrollo de un módulo para reportería con Dapper, porque proporciona una solución innovadora en comparación con otros ORM (Object-Relational Mapping). Dapper es una tecnología que permite manipular bases de datos mediante consultas SQL con el fin de maximizar los recursos y mejorar el rendimiento. A través de Dapper, las consultas Sql en Microsoft SQL Server se llevaron a cabo utilizando cadenas de conexión previamente definidas. Además, se utilizó una API REST para comunicarse con el sistema y el módulo de reportería, permitiendo una conexión segura a los datos. Esta implementación permite a los analistas crear informes basados en sus necesidades personalizando la información a través de filtros. De esta manera, se fortalece el compromiso institucional con la innovación y la mejora continua

En la actualidad el Sistema de Gestión de la Investigación tiene limitaciones para recopilar y analizar datos que sean relevantes sobre los proyectos de investigación de la Universidad Nacional de Chimborazo. Por esta razón implementar un módulo para generar reportes es una solución que ayuda a optimizar el tiempo sobre estos procesos. Este módulo permite a los analistas acceder a la información de una forma más rápida y clara sobre el estado y avance de los proyectos.

El proyecto de investigación tiene como objetivo mejorar significativamente la accesibilidad y el análisis de datos a través de la implementación de una metodología para generar informes utilizando Dapper y servicios web API REST. Esta metodología proporciona herramientas innovadoras para la consulta y el análisis de datos, promoviendo así la eficiencia y la transparencia en la gestión académica de la UNACH.

1.1 Planteamiento del Problema

El departamento de investigación de la Universidad de Chimborazo no dispone de un sistema específico para la revisión de trabajos de investigación. Esto limita la accesibilidad de datos significativos para los analistas en diversos proyectos de investigación. La ausencia de una estructura adecuada para la creación de datos colaborativos entre los investigadores y el Director de Investigación resulta poco eficaz. Dentro del alcance de la investigación de la UNCACH, existe la necesidad de múltiples herramientas para facilitar la recuperación de información para comparaciones de subclases y avanzar hacia indicadores objetivos

presentados de manera abierta y clara tanto a la academia como a los organismos de gobierno; la recuperación de datos necesita ser altamente eficiente.

1.2 Justificación

La Universidad Nacional de Chimborazo ha establecido el Sistema de Gestión de la Investigación, el cual permite gestionar documentos y asignar recursos a los proyectos como optimizar los informes mediante análisis predictivos. Cada proyecto cuenta con el apoyo de un analista quien elabora un informe que evalúa el estado de avance del proyecto activo. No obstante, la Información que es generada por la unidad de información permite mayores controles y monitoreos. Con la ausencia de eficiencia, agilidad y un alto control en la transparencia dicha metodología se complica en su funcionalidad. En un Sistema de Gestión de la Investigación la falta de metodologías precisas para la elaboración de informes restringe el acceso a información precisa bloqueando la posibilidad de que los aliados generales e investigadores tomaran decisiones estratégicas importantes que optimen el desarrollo académico e investigativo en una realidad internacional en la que se requiere adaptabilidad y flexibilidad.

El motivo para este proyecto es atender las necesidades de los investigadores de la UNACH. Se planteó el desarrollo de un módulo de reportes junto con la integración de Dapper con servicios web API REST que permitirán a los analistas de proyectos consultar y realizar análisis de datos complejos de forma casi instantánea sobre los datos referentes a los proyectos de investigación. Proporcionar esta herramienta se espera mejorar la investigación en la UNACH y el impacto que esta tiene en la comunidad científica internacional, al reducir los tiempos en los que se necesita acceder a la información, lo cual, optimiza la lógica científica.

1.3 Formulación del Problema

¿Cómo la integración de Dapper mediante servicios web API REST mejora la eficiencia de los reportes en el sistema de gestión de Investigación de la UNACH?

1.4 Objetivos

Objetivo General

Integrar Dapper para generar reportes en el Sistema de Gestión de Investigación de la Universidad Nacional de Chimborazo

Objetivos Específicos

- Realizar una revisión sistemática de literatura sobre tecnologías y estándares aplicables en la integración de la biblioteca Dapper y servicios web API REST.
- Implementar un módulo de generación de reportes dentro del sistema de gestión de Investigación existente, utilizando Dapper y servicios web API REST.
- Evaluar la eficiencia del módulo de generación de reportes desarrollado

CAPÍTULO II. MARCO TEÓRICO

2.1 Sistema de gestión de investigación de la Universidad Nacional de Chimborazo

Es una plataforma creada para administrar los proyectos de investigación en el ámbito universitario. La Dirección de Investigación es la encargada del sistema, la cual supervisa y autoriza todos los proyectos. La meta principal es apoyar a alumnos e investigadores en la realización de sus proyectos de investigación [1]. La figura 1 presenta el sistema a continuación.

Figura 1: Sistema de gestión de la investigación

2.2 API REST

Se trata de una arquitectura vanguardista creada para simplificar la comunicación e interacción entre distintos sistemas mediante la red. Emplea técnicas HTTP, tales como GET, POST, PUT y DELETE, para el uso de recursos, además de administrar eficazmente las peticiones y reacciones. Este método posibilita que aplicaciones o sistemas externos adquieran acceso a información concreta, características de un servicio o aplicación, incluyendo el acceso regulado a bases de datos. Esto se lleva a cabo bajo supervisión y conforme a normas previamente fijadas para asegurar la seguridad y fiabilidad del intercambio de datos [2].

2.2.1 Características de API REST

Para asegurar un diseño eficaz, las APIs REST necesitan satisfacer ciertos requisitos esenciales que faciliten una comunicación nítida, organizada y optimizada entre el cliente y el servidor, garantizando que las peticiones y respuestas se gestionen de manera adecuada. Además, la implementación de estos principios promueve la colaboración entre sistemas, optimizando el desempeño a través del almacenamiento en caché y posibilitando una arquitectura adaptable y segura [2]. A continuación, en la tabla 1 se detallan los atributos considerados relevantes para establecer una API REST.

Tabla 1. API REST Características

Características	descripción
Arquitectura cliente-servidor	El cliente envía una solicitud al servidor y el servidor responde.
Sin estado (Stateless)	Cada solicitud incluye toda la información requerida para procesarla.
Cacheable	Las respuestas muestran si se pueden almacenar en una memoria caché.
Interfaz uniforme	Procesos como GET, POST, PUT y DELETE; formatos como JSON, XML.
Sistema en capas	Creado en capas para mejorar la gestión y la escalabilidad.
Código bajo demanda	En lugar de datos, el servidor puede enviar un código para que el cliente lo ejecute.

Fuente: Adaptado de [2]

2.2.2 Ventajas de API REST

El uso de APIs RESTful ha crecido significativamente en los últimos años, convirtiéndose en un estándar para la arquitectura de software, y muchos desarrolladores las implementan actualmente, optimizando su interoperabilidad y eficiencia [3], como se especifica a continuación.

- **Ligero:** Usa el estándar HTTP, que permite trabajar con diferentes formatos, conteniendo HTML, JSON y XML.
- **Independiente:** Libertad de trabajar de forma autónoma, explorando varias áreas del proyecto.
- **Escalable y flexible:** Crecimiento rápido gracias a la dispersión cliente-servidor.

2.3 Dapper

Dapper es un micro ORM hecho para aplicaciones .NET que optimiza la comunicación entre una aplicación y una base de datos, mejorando la ejecución de consultas SQL. Fue desarrollado por el equipo de Stack Overflow [4]. La compañía solicitaba una herramienta que pudiera manipular datos masivos de manera sencilla y veloz. Dapper no solo simplifica el trabajo con los datos, sino que también optimiza la velocidad y efectividad de las consultas, lo cual es fundamental para la eficiencia de las aplicaciones. En la figura 2 se presenta la interacción de una base de datos con Dapper.

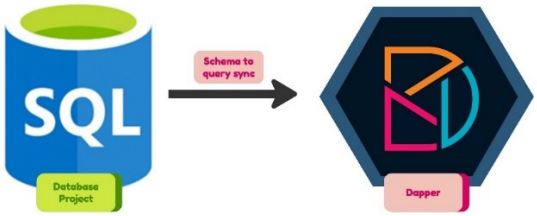


Figura 2: Dapper
Fuente: [5]

2.3.1 Características de Dapper

Dapper opera de manera directa con consultas SQL, proporcionando un control total sobre su realización. Su sencillez al vincular resultados a objetos. NET lo convierte en la elección perfecta para aplicaciones que requieren alto rendimiento y baja sobrecarga. Además, Dapper soporta consultas SQL complejas y es altamente compatible con una variedad de bases de datos, lo que lo convierte en una herramienta versátil y poderosa para la interacción rápida de datos [6]. Entre las características de Dapper se localizan:

- Velocidad y rapidez en el rendimiento.
- Menos líneas de código.
- Fácil manejo de consultas SQL.
- Acceso directo a la base de datos.
- Soporte en diversas consultas.
- Funcionalidad de inserción de datos masivos.

2.3.2 Lenguajes que utiliza Dapper

Dapper fue diseñado principalmente para ser utilizado con C#, el lenguaje más popular en .NET; sin embargo, también se puede usar con otros lenguajes que componen esta plataforma, como VB.NET y F#. Dapper se puede utilizar de manera eficiente en aplicaciones escritas en cualquiera de estos lenguajes debido a su integración con .NET. Esto facilita el trabajo de los desarrolladores que necesitan trabajar con bases de datos en sus proyectos [7].

2.3.3 Integración de Dapper y API REST

Con Dapper, vincular consultas SQL directamente con .NET simplifica el acceso a la base de datos y optimiza el rendimiento de las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) en los servicios API REST. Una API REST procede como un intermediario entre el cliente y el servidor, gestionando las peticiones HTTP, mientras que Dapper tiene la tarea de entrar y cambiar los datos en la base. La combinación de ambos garantiza que las operaciones sean veloces y livianas, proporcionando una respuesta efectiva[8]. A continuación, se aprecia los endpoints en la figura 3.

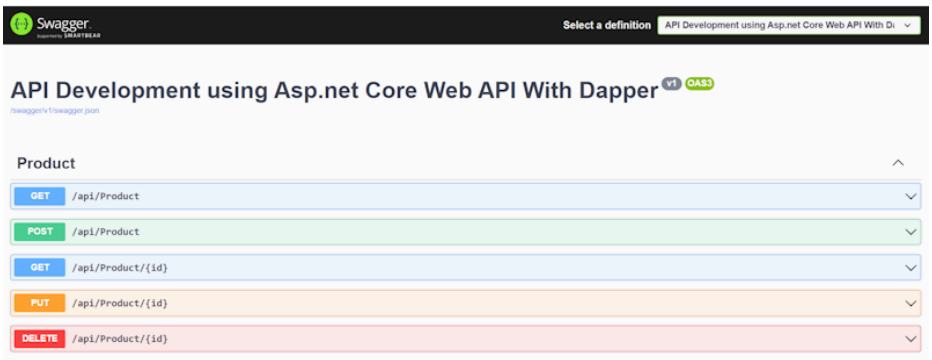


Figura 3: Dapper y API REST
Fuente: [8]

2.3.4 Ventajas y desventajas de Dapper

El objetivo principal de Dapper es proporcionar un acceso rápido y efectivo a las bases de datos para que los desarrolladores puedan escribir consultas SQL directamente en el código. No obstante, como cualquier tecnología, su adopción debe evaluarse cuidadosamente. Por ello, es fundamental analizar tanto sus ventajas como sus desventajas para determinar su integración en un proyecto. En la Tabla 2 se presentan algunos de los principales beneficios e inconvenientes asociados al uso de Dapper.

Tabla 2. Ventajas y desventajas de Dapper

Ventajas	Desventajas
Dapper es uno de los ORM más rápidos dentro del mundo .NET.	Dapper requiere escribir consultas SQL manualmente.
Permite escribir menos líneas de código en comparación con otros ORM.	Proporciona menos abstracción que otros ORM, que pueden requerir más trabajo manual
Soporta múltiples bases de datos y consultas SQL complejas.	Aunque es popular, no se encuentra mucha información para usarlo.
Soporta el uso de procedimientos almacenados.	

Fuente: Adaptado de [9]

2.4 Revisión sistemática sobre el uso de Dapper con servicios API REST

Se realizó una revisión sistemática de la literatura con el objetivo de analizar las tecnologías y estándares aplicables en la integración de Dapper con servicios web API REST. Esta revisión se orientó a identificar experiencias previas, evaluar resultados técnicos y fundamentar la elección de tecnologías para la implementación del módulo de reportes en el sistema de gestión de investigación. La estrategia de búsqueda se realizó en fuentes académicas reconocidas como Google Scholar y repositorios institucionales. Se utilizaron combinaciones de palabras clave tales como: “Dapper ORM”, “API REST .NET” y “Dapper vs Entity Framework”, acotando los resultados a publicaciones comprendidas entre los años 2019 y 2025.

Los criterios de inclusión establecidos para la selección de estudios fueron: investigaciones que aplican Dapper como micro ORM, integración con servicios API REST, evaluaciones de rendimiento bajo métricas técnicas o estándares de calidad del software, especialmente en entornos .NET. Por otro lado, se excluyeron artículos que no presentaban una aplicación práctica, estudios sin evaluación técnica o publicaciones de tipo divulgativo sin respaldo metodológico. Como resultado de la búsqueda, se seleccionaron dos estudios relevantes que cumplen con los criterios establecidos. El primero corresponde a un proyecto desarrollado en la Universidad Nacional de Chimborazo titulado “Análisis del Desempeño entre Dapper y Entity Framework: Caso Aplicativo en el Sistema de Buenas Prácticas en Empresas Turísticas de Riobamba” [10].

En dicho estudio se utilizó Dapper y Entity Framework como tecnologías comparativas en la capa de acceso a datos de un sistema implementado con API REST. La evaluación del desempeño se basó en la norma ISO/IEC 25010, enfocándose en los atributos de comportamiento temporal, uso de recursos y capacidad. Se utilizaron herramientas como JMeter con la extensión PerfMon y la clase System.Diagnostics de .NET, mediante los métodos TimeSpan y PerformanceCounter, para registrar los tiempos de respuesta y consumo de recursos. Los resultados demostraron que Dapper ofrece una mejora significativa en el rendimiento con respecto a Entity Framework, reduciendo los tiempos de respuesta y el uso de memoria del sistema.

El segundo estudio es "Análisis del rendimiento de las herramientas de mapeo objetorelacional (ORM) en el entorno .Net 6" de Darshan Patel et al. [11]. Esta investigación evalúa las tecnologías ORM Dapper, Entity Framework Core y NHibernate en función del rendimiento de consultas, consumo de CPU y memoria. La investigación concluyó que Dapper ORM proporciona una eficiencia óptima en todas las operaciones de tiempo de ejecución, tanto simples como complejas, con un menor consumo de recursos y tiempos de respuesta más rápidos que las otras tecnologías evaluadas.

Como conclusión, los dos estudios analizados coinciden en validar que Dapper es una tecnología eficiente para contextos de alto entorno demanda en el rendimiento del acceso a datos. Su combinación con servicios Web API REST en el entorno .NET hace de Dapper una alternativa ágil, flexible y menos pesada que otros ORMs. Esta evidencia técnica justifica su implementación en el sistema de gestión de investigación de la Universidad Nacional de Chimborazo con la intención de utilizarlo como base para construir un módulo de reportes responsivo y eficiente, en términos de recursos. A continuación, se presenta la Tabla 3 que resume y contrasta los aspectos analizados en los estudios incluidos en esta revisión sistemática.

Tabla 3. Revisión Sistemática

Estudio	Tecnología Evaluada	Métricas ISO/IEC 25010	Herramientas de Prueba	Resultado
Universidad Nacional de Chimborazo	Dapper vs Entity Framework (.NET + API REST)	Comportamiento temporal, uso de recursos, capacidad	JMeter + PerfMon, System.Diagnostics (TimeSpan, PerformanceCounter)	Dapper mejoró notablemente los tiempos de respuesta y uso de memoria
Darshan Patel et al.	Dapper vs EF Core vs NHibernate (.NET 6 + API REST)	Tiempo de ejecución, consumo de CPU y memoria	Scripts de benchmarking en .NET 6	Dapper presentó el mejor rendimiento en operaciones simples y complejas

Fuente: Adaptado de [10], [11]

2.5 Herramientas de desarrollo

Las herramientas de desarrollo ayudan a simplificar la creación, mantenimiento y gestión de aplicaciones. Al proporcionar un entorno que estructura los procesos de codificación, depuración y pruebas, sin estas herramientas el desarrollo de software enfrentará problemas como la corrección de errores, lo que a su vez impedirá la productividad y la calidad.

2.5.1 JetBrains Rider

Es un IDE destinado al desarrollo tanto de frontend como de backend, principalmente orientado a.NET. Su extenso abanico de funciones facilita la creación de aplicaciones centradas en.NET, ASP.NET Core, MAUI, o en motores de juegos como Unity, Unreal Engine o Godot. Su estructura está concebida para ofrecer rapidez y capacidad de reacción. Además, es un IDE multiplataforma ya que ofrece una experiencia de desarrollo superior en Windows, macOS y Linux al convertir las instrucciones de observaciones de código que soliciten grandes recursos en un proceso autónomo y eliminar de manera eficiente las interrupciones como los bloques en la interfaz de usuario y las retracciones en la entrada de texto [12]. A continuación, en la figura 4 se muestra el IDE de Rider.

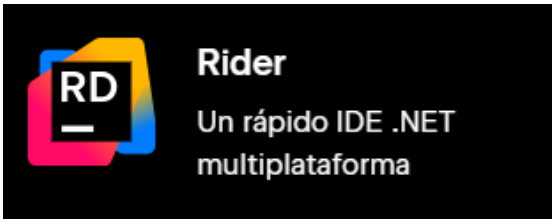


Figura 4: JetBrains Rider
Fuente: [8]

2.5.2 SQL Server Management Studio (SSMS)

SQL Server Management Studio (SSMS), un software destinado a la gestión de la plataforma de bases de datos de Microsoft SQL Server, facilita a los usuarios el acceso, la configuración, la administración y la creación de componentes de SQL Server, Azure SQL Database, Azure SQL Managed Instance y Azure Synapse Analytics. Suministra un entorno unificado para el manejo eficiente de cualquier infraestructura SQL [13], optimizando la productividad con funciones como la ejecución y culminación de consultas, la administración de la seguridad y el monitoreo del desempeño mediante una interfaz gráfica intuitiva, tal como se ilustra en la figura 5.

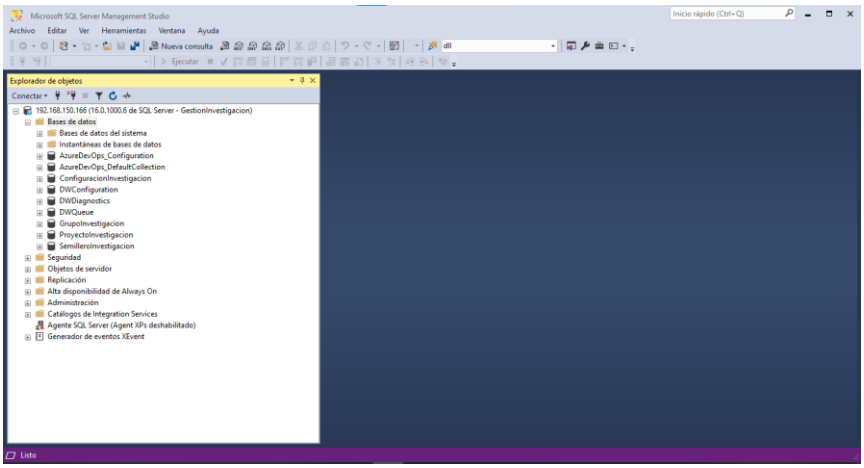


Figura 5: Microsoft SQL Server

2.5.3 QuestPDF

Es una librería de C# que adopta un enfoque revolucionario para la generación de PDF, ofreciendo un motor de diseño diseñado y optimizado específicamente para generar PDF en vez de confiar en HTML como lo hacen numerosas otras soluciones. Opera fusionando componentes como texto, imágenes, tablas y hojas de cálculo en estructuras más sofisticadas [14].

2.5.4 OfficeOpenXml

El formato de archivo Office Open XML (OOXML) es un estándar basado en XML desarrollado por Microsoft para la representación estructurada de documentos de procesamiento de texto, hojas de cálculo y presentaciones [15]. Se encuentra normalizado por ECMA International bajo la especificación ECMA-376 y por ISO/IEC mediante la norma ISO/IEC 29500.

2.6 Norma ISO 25010

La norma ISO/IEC 25010 se concentra en la valoración de la calidad del software y los sistemas. Esta norma internacional ofrece una guía para evaluar los requerimientos de calidad que deben cumplir un producto de software, con el fin de que su utilización por parte del usuario sea satisfactoria y eficiente [16]. La calidad del software ISO/IEC 25010 se divide en las dos subpartes principales: la calidad del producto y la usabilidad. Ambas partes incluyen características y subcaracterísticas que permiten evaluar dicho producto [17].

Según [18], el modelo de calidad es un dispositivo crítico para apreciar un producto de software ya que establece las características fundamentales que determinan si el producto cumple con los requisitos de sus usuarios y, por ende, agrega valor. Este modelo permite el análisis de aspectos como la capacidad de servicio, funcionalidad, rendimiento, seguridad y mantenibilidad a través de la clasificación de la calidad en características y subcaracterísticas. Como se ilustra en la figura 6, la norma ISO/IEC 25010 define nueve características esenciales que se centran en esta evaluación para asegurar que el producto cumpla efectivamente con las expectativas del usuario.

Figura 6: Norma ISO/IEC 25010: Calidad del software
Fuente: [18]

Estos estándares brindan a las empresas y a los desarrolladores una base consistente para implementar mejoras continuas en sus productos y, al adherirse a ellos, será posible no solo compensar las expectativas y necesidades de los usuarios, de igual forma responder de manera efectiva a las solicitudes cambiantes del mercado. Esto favorece significativamente al desarrollo de productos más confiables, aumentando la lealtad y satisfacción del cliente, y fortaleciendo la competitividad de la organización en el mercado actual

2.6.1 Eficiencia de Desempeño

Uno de los aspectos más concluyentes del modelo de calidad del producto de software es su capacidad para proporcionar un nivel adecuado de rendimiento basado en la cantidad de los recursos utilizados en condiciones especificadas, lo que significa que el software debe realizar sus funciones dentro de los parámetros de tiempo y rendimiento especificados, al tiempo que utiliza de manera eficiente recursos como la CPU, la memoria, el almacenamiento y la energía bajo ciertas condiciones de uso [18].

Este aspecto, denominado eficiencia de rendimiento, se desglosa en varios subaspectos que permiten una evaluación más completa de su funcionamiento:

- Comportamiento temporal: Grado en el que el producto logra sus objetivos dentro del tiempo de respuesta y nivel de output especificado durante el marco de tiempo delineado
- Utilización de recursos: Analiza si la cantidad y el tipo de recursos utilizados por el software cuando realiza sus funciones en condiciones específicas se mantienen dentro de los límites especificados.
- Capacidad: Examina si existe el software que satisface los requisitos en relación a los límites máximos a ciertos criterios que se consideran de gran importancia.

Estas subcaracterísticas apoyan a valorar el rendimiento del software en proporción con sus requerimientos, asegurando una funcionalidad adecuada sin exceder todos los recursos asignados [18].

2.7 Metodología Kanban

Kanban es una metodología basada en una filosofía de mejora constante, en la que las tareas se "extraen" de un listado de tareas por realizar en un flujo constante de trabajo. Se lleva a cabo mediante tablas Kanban, una técnica visual de administración de proyectos que facilita a los equipos la visualización de su flujo laboral y volumen de trabajo. Esto asiste a los equipos en la búsqueda de un balance entre la labor que requieren realizar y la disponibilidad de cada integrante del equipo. En una tabla Kanban, el proyecto se presenta como una tabla con columnas que simbolizan tradicionalmente una etapa de trabajo. La tabla Kanban más elemental puede contener columnas como "Trabajo en curso", "Trabajo pendiente" y "Terminado" [19].

CAPÍTULO III. METODOLOGIA

3.1 Tipo de Investigación

El estudio se llevó a cabo bajo un enfoque cuantitativo, y se analizó el caso de Dapper en la empresa utilizando la norma ISO 25010, que evalúa la calidad de los productos de software. Se calcularon, evaluaron y compararon las siguientes métricas: tiempo y recursos, así como capacidad. Estas conclusiones mostraron si la incorporación de Dapper optimiza el sistema de gestión en el nivel operativo del sistema investigacional y si se cumplen las condiciones de calidad requeridas.

3.2 Diseño de la Investigación

Las variables no fueron alteradas directamente, el estudio no fue experimental; En cambio, fueron observados en su entorno natural, libres de interferencias externas. Este enfoque permitió la recolección de datos objetivos y realistas, asegurando la validez de los resultados obtenidos y proporcionando una visión auténtica del comportamiento de las variables en estudio.

3.3 Población de Estudio y Tamaño Muestra

En este estudio se consideró que la población era infinita debido al análisis de múltiples indicadores referente al desempeño del módulo reportes del Sistema de Gestión de la Investigación JMeter, así como al monitoreo del sistema. La evaluación contempló la capacidad del sistema, la gestión de los recursos y el comportamiento temporal o del sistema, por lo que no fue necesario fijar un tamaño muestral.

3.4 Técnicas de Recolección de Datos

Se empleó Apache JMeter como instrumento de recolección y estudio de datos para propósitos de instrumentación.

3.5 Métodos de Análisis y Procesamiento de Datos

El desempeño del sistema fue medido y analizado con base en los requerimientos de la norma ISO/IEC 25010 usando Apache JMeter, lo cual facilitó la cuantificación del comportamiento temporal, del consumo de recursos y de la potencia de procesamiento todo gracias a Dapper. Los datos obtenidos se examinaron de manera estadística para establecer la presencia de variaciones importantes en el rendimiento del sistema, proporcionando de esta manera un criterio imparcial acorde a los estándares ISO / IEC 25010.

3.6 Identificación de variables

3.6.1 Variable dependiente

La eficiencia de los reportes en el sistema de gestión de investigación de la UNACH

3.6.2 Variable independiente

Integración de Dapper para la generación de reportes mediante servicios web API REST

3.7 Operacionalización de variables

A continuación, en la Tabla 4 se muestra cómo se operacionalizan las variables

Tabla 4. Operacionalización de variables

PROBLEMA	TEMA	OBJETIVOS	VARIABLES	CONCEPTUALZACION	DIMENSION	INDICADORES
Como la integración de Dapper mediante servicios web API REST mejora la eficiencia de los reportes en el sistema de gestión de Investigación de la UNACH	Integración de Dapper para generar reportes en el sistema de gestión de investigación de la Universidad Nacional de Chimborazo	GENERAL Integrar Dapper para la generación de reportes mediante servicios web API REST en el Sistema de Gestión de Investigación de la Universidad Nacional de Chimborazo	INDEPENDIENTE Integración de Dapper para la generación de reportes mediante servicios web API REST	Dapper es una biblioteca de código abierto para Microsoft .NET Framework que facilita el acceso a bases de datos SQL, consultas SQL. Los servicios REST web API son una interfaz de programación de aplicaciones (API) fundamentada en la arquitectura REST, una arquitectura que define la manera en que se pueden compartir datos entre sistemas web. Los servicios de API web REST generalmente hacen uso del protocolo HTTP para el envío y recepción de información.	Integración de Dapper utilizando servicio web API REST	Independiente. • Tiempo de Ejecución de Consultas • Número de Reportes Generados por minuto.
		ESPECIFICOS • Realizar una revisión sistemática de literatura sobre tecnologías y estándares aplicables en la integración de la biblioteca Dapper y servicios web API REST. • Implementar un módulo de generación de reportes dentro del sistema de gestión de investigación existente, utilizando la biblioteca Dapper y servicios web API REST. • Evaluar la eficiencia del módulo de generación de reportes desarrollado.	DEPENDIENTE La eficiencia de los reportes en el sistema de gestión de investigación de la UNACH	Una norma internacional denominada ISO 25010 proporciona una serie de atributos de calidad para la valoración de software, y entre estos atributos, la eficiencia tiene un rol fundamental. Hace referencia a la habilidad del software para llevar a cabo sus funciones en términos de tiempo y rendimiento determinados, mientras utiliza de manera eficaz los recursos existentes.	Calidad del Software	Dependiente. • Comportamiento temporal • Utilización de recursos • Capacidad

3.8 Metodología de Desarrollo

En este proyecto de investigación, se utilizó la metodología ágil Kanban para gestionar de forma visual y ordenada la integración de Dapper para la generación de reportes de los proyectos de investigación de la Universidad Nacional de Chimborazo. La metodología Kanban se centra en la gestión del flujo de trabajo mediante tableros que permiten visualizar el estado de cada tarea en columnas. Para una mejor comprensión el proyecto se dividió en 5 fases: inicio, diseño, implementación, lanzamientos y pruebas. Estas fases permitieron gestionar el avance de las tareas de manera eficaz y transparente, asegurando que cada período del proceso de integración de Dapper se llevara a cabo de manera ordenada y transparente.

3.8.1 Inicio

Durante esta fase se definieron los requerimientos y tareas iniciales para la integración de Dapper al Sistema de Gestión de la Investigación de la Universidad Nacional de Chimborazo. A continuación, en la Tabla 5, se muestra un diseño inicial de la tabla Kanban para el desarrollo del módulo:

Tabla 5. Diseño inicial

Por hacer	En proceso	En revisión	Completado
Levantar requerimientos funcionales y no funcionales.	Realizar entrevistas con el personal a cargo del Sistema de gestión de la investigación de la Universidad Nacional de Chimborazo	Estudiar las herramientas necesarias para implementar Dapper y generar reportes.	
Definir la estructura de datos para los reportes.			
Instalación y configuración de JetBrains Rider 2024.1.1.			
Instalar SQL Server para configurar las rúbricas..			
Instalación de Dapper y configuración de la conexión a SQL Server.			
Implementación de la lógica para consulta de reportes.			
Pruebas de funcionalidad			
Corregir posibles errores			

Especificación de requerimientos funcionales

La Tabla 6, demuestra los requisitos funcionales para la integración de Dapper en el Sistema de Gestión de la Investigación de la Universidad Nacional de Chimborazo. Estos requisitos

describen las funciones específicas que debe cumplir el módulo para satisfacer las necesidades del usuario y alcanzar los objetivos del proyecto.
Cada solicitud se identifica mediante un ID para una mejor comprensión. Las solicitudes se obtuvieron luego de una entrevista con el personal responsable del sistema de gestión de investigaciones de la UNACH.

Tabla 6. Requisitos funcionales

ID	Nombre del Requerimiento	Descripción	Prioridad
RF01	Generación de Reportes	El usuario podrá generar reportes en formato PDF y Excel	Alta
RF02	Filtrado de Datos	El usuario podrá aplicar filtros (por facultad, convocatoria, estado del proyecto) antes de generar un reporte.	Alta
RF03	Pre visualización de Resultados PDF	El usuario podrá visualizar un resumen de resultados antes de la descarga.	Alta
RF04	Descarga Automática en Excel	Si el usuario selecciona Excel para ver el reporte empezará la descarga.	
RF05	Validación de Datos	El sistema verificará la existencia de datos antes de permitir la generación de un reporte.	
RF06	Conversión de Formato	El sistema convertirá los reportes generados a formato Base64 para su transferencia y visualización.	

Especificación de requerimientos no funcionales

La Tabla 7 muestra el enfoque sistemático utilizado para resaltar los requisitos no funcionales del proyecto, junto con los aspectos funcionales del módulo en áreas como: eficiencia, seguridad, tecnología, entre otros. Cada uno de los requisitos no funcionales brinda a la implementación una Dirección para el desarrollo y de ser ejecutado, complementar la experiencia con un funcionamiento confiable, eficaz, y optimizado en el resultado.

Tabla 7. Requerimientos no funcionales

ID	Categoría	Requerimiento	Descripción
RNF01	Eficiencia	Tiempo de Respuesta	Los reportes deben generarse en formato PDF y Excel en un tiempo máximo de 5 segundos bajo carga normal.

Especificación de requerimientos no funcionales técnico

En la Tabla 8, los requisitos específicos no funcionales del módulo de informes del Sistema de Gestión de la Investigación de la Universidad Nacional de Chimborazo. Estos requisitos se basan en la efectividad del sistema en sus actividades críticas tales como comportamiento temporal, utilización de recursos y capacidades de desconexión de carga del sistema.

Tabla 8. Requerimientos no funcionales técnicos

ID	Categoría	Requerimiento	Descripción
RNF01	Eficiencia	Tiempo de Respuesta	Los reportes deben generarse en un tiempo máximo de 5 segundos bajo carga normal.
RNF02	Eficiencia	Comportamiento Temporal	El sistema debe responder de manera rápida y eficiente a las solicitudes de generación de reportes, sin que el tiempo de respuesta aumente significativamente con el número de registros.
RNF03	Eficiencia	Utilización de Recursos	El sistema debe utilizar eficientemente los recursos del servidor, evitando el consumo excesivo de memoria y CPU durante la generación de reportes.
RNF04	Eficiencia	Capacidad	El sistema debe ser idóneo para generar reportes para al menos 4 registros paralelamente sin complicar el rendimiento.

3.8.2 Diseño

Diagrama de Casos de uso

A continuación, la figura 7 demuestra como interactúa el analista con el sistema de gestión de la investigación para generar un reporte.

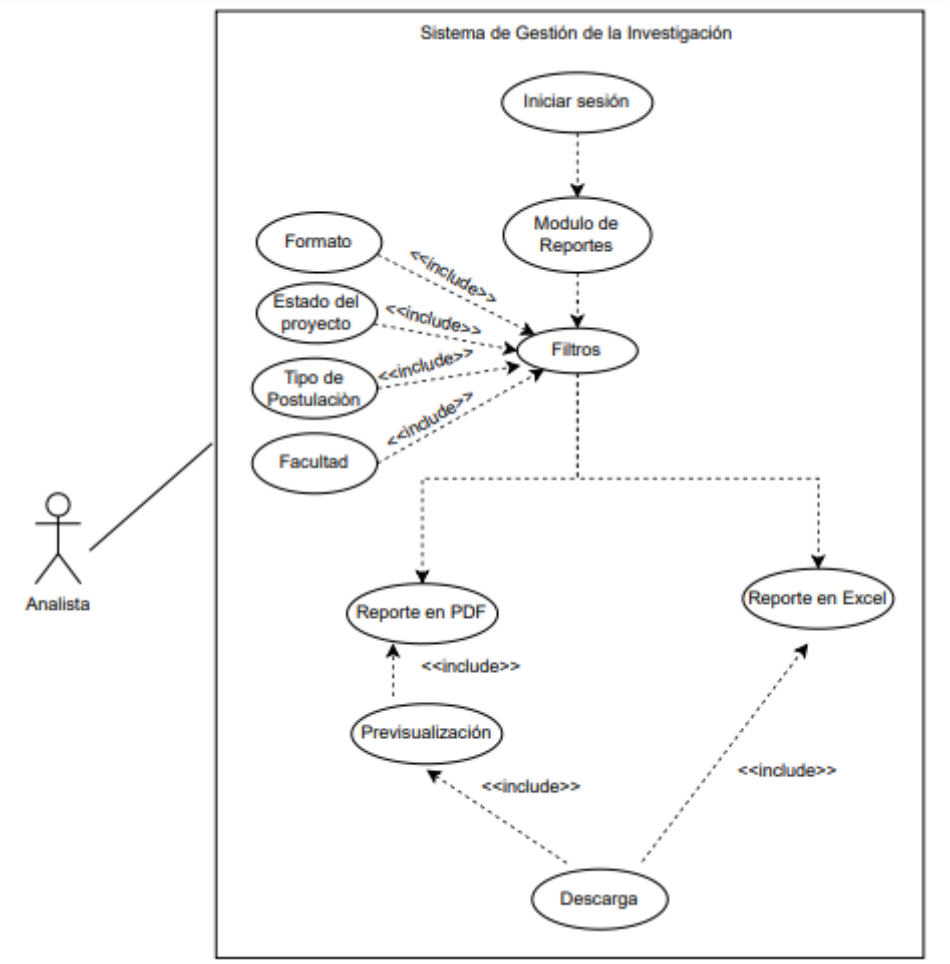


Figura 7: Generación de reportes

Arquitectura cliente servidor del modulo

La arquitectura seleccionada para el diseño del módulo se especificó como clienteservidor, todas las solicitudes ejecutadas a través del módulo son resueltas por el servidor. SQL Server Management Studio se utilizó como servidor para este proyecto, lo que facilitó una gestión eficaz y segura de la base de datos en tiempo real. A continuación, la figura 8 presenta un esquema que ilustra la forma en que estos elementos se incorporaron en el sistema.

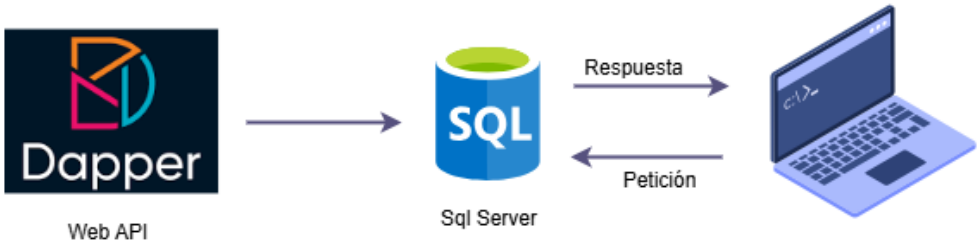


Figura 8: Arquitectura

Actualización del Tablero Kanban

Durante el desarrollo del módulo de reportes, se actualizó el tablero Kanban para reflejar el avance de las actividades planificadas. La Tabla 9 muestra un ejemplo del estado intermedio del tablero Kanban, donde se evidencia la evolución del flujo de trabajo del proyecto.

Tabla 9. Tablero Kanban, fase Implementación

Por hacer	En proceso	En revisión	Completado
Pruebas de rendimiento con alta carga Generación de reportes en Base64	Implementación de filtros por convocatoria Desarrollo del endpoint /project-report/project-by-calls	Validación de resultados en PDF	Instalación de herramientas (Rider, SQL) Configuración de conexión con Dapper Validaciones básicas de los datos

3.8.3 Implementación

l primer paso en el desarrollo del módulo de reportes, fue crear un nuevo proyecto de Web Api en Jetbrains Rider, que es una interfaz que permite la comunicación con otras aplicaciones, proporcionando flexibilidad, reusabilidad y escalabilidad en el desarrollo de aplicaciones. La figura 9 muestra la creación del proyecto.

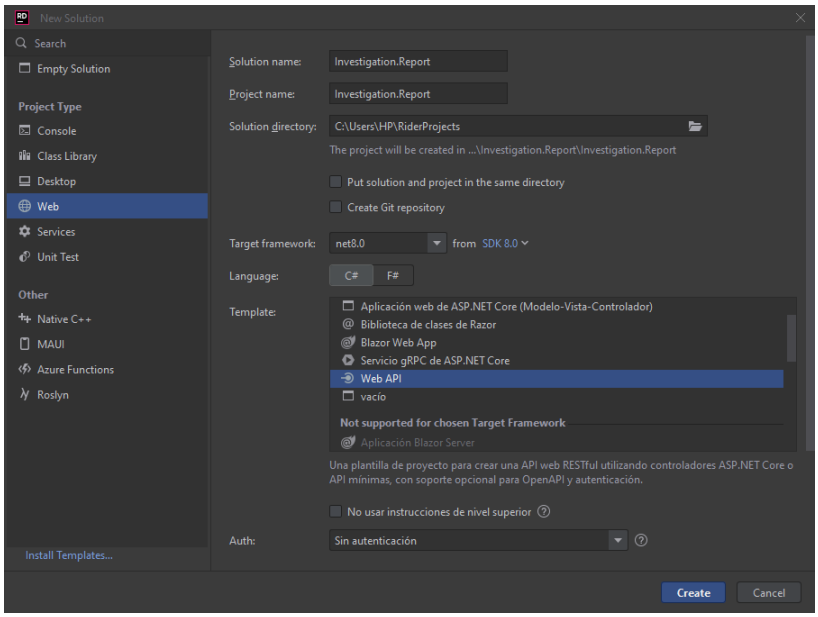


Figura 9: New Solution

Posteriormente, se implementaron cuatro capas clave que ayudaron con la organización del proyecto; cada capa está planteada para desempeñar un trabajo específico dentro de la arquitectura:

- **Investigation.Report.Api:** Simboliza la capa de API, encargada de mostrar los endpoints REST.
- **Investigation.Report.Application**: Encargado de la lógica de aplicación, interviniendo entre la API y las capas de dominio e infraestructura.
- **Investigation.Report.Domain:** Delimita los modelos, entidades y contratos que establecen las reglas del negocio y la comunicación entre capas.

- **Investigation.Report.Infrastructure:** Trata la interacción con la base de datos y otros servicios externos, constituyendo herramientas como Dapper para la ejecución de consultas SQL.

Además, la figura 10 ilustra cómo se estructuraron los límites máximos.

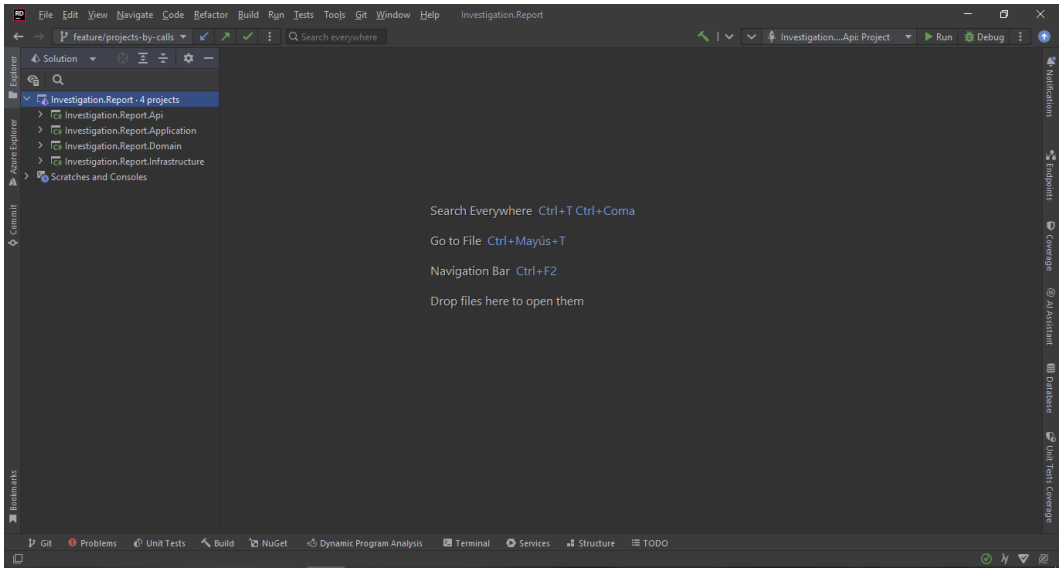


Figura 10: Capas principales

A continuación, la descarga de Dapper se llevó a cabo mediante los paquetes NuGet, como se muestra en la figura 11. Estos paquetes permiten la instalación, actualización y gestión de aquellas bibliotecas que se requieran de forma sencilla.

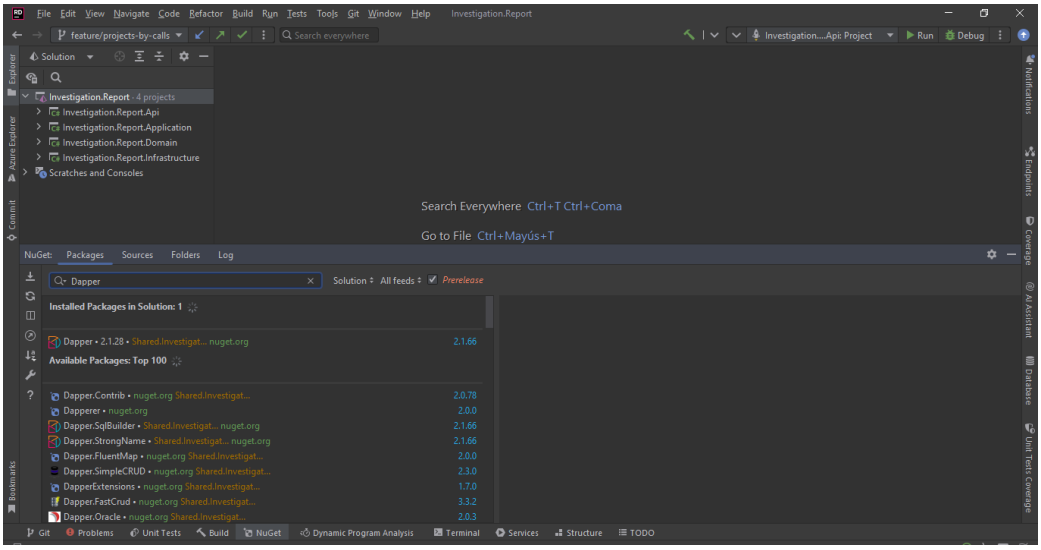


Figura 11: Dapper descarga

Para conectarse a la Base de Datos, la configuración de la aplicación. Development.json archivo se utilizó para configurar las cadenas de conexión y otros parámetros requeridos en el entorno de desarrollo del proyecto, como se muestra en la figura 12.

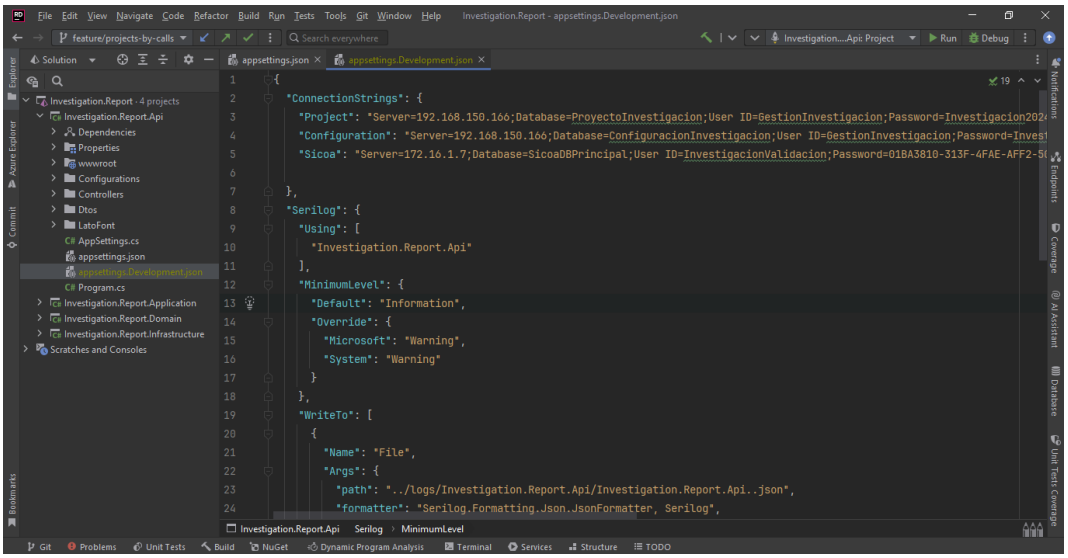


Figura 12: Conexión

La capa Investigation.Report.Domain es la administradora de las entidades, reglas de negocio. Representa claramente el modelo de dominio como su objetivo principal y mantiene la lógica de negocio central que se muestra en la Figura 13, permitiendo que otras empresas interactúen con él a través de contratos o interfaces sin comprometer su independencia.

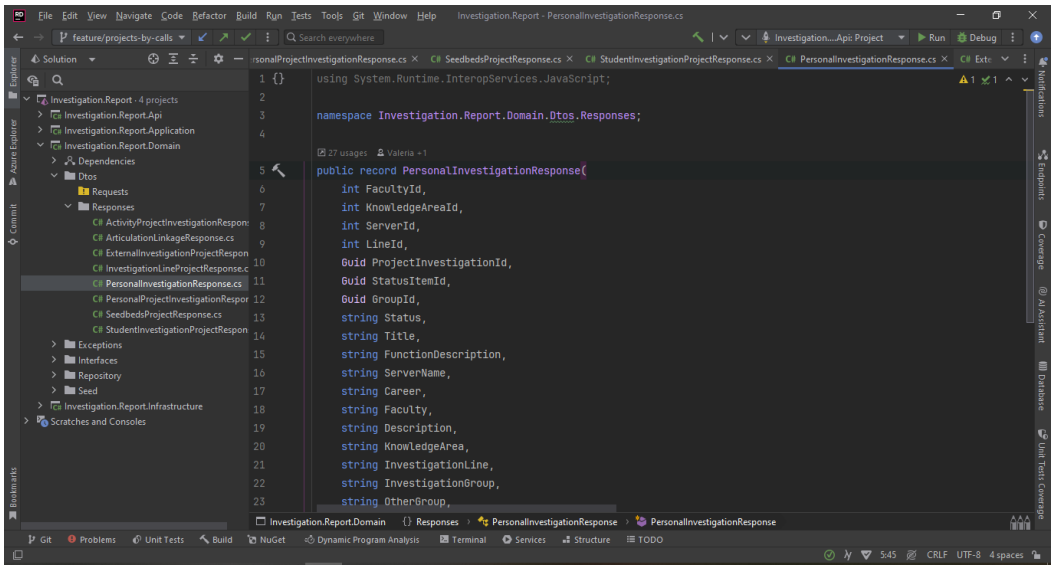


Figura 13: Capa de dominio

En archivo ProjectReportService.cs fue creado en la capa Investigation.Report.Infraestructure, donde se llevó a cabo la lógica necesaria para producir informes dentro del proyecto. El archivo ProjectRepository.cs ubicado en la carpeta Repository tramita la interacción con la base de datos por medio de consultas sql, tal como se ilustra en la figura 14.

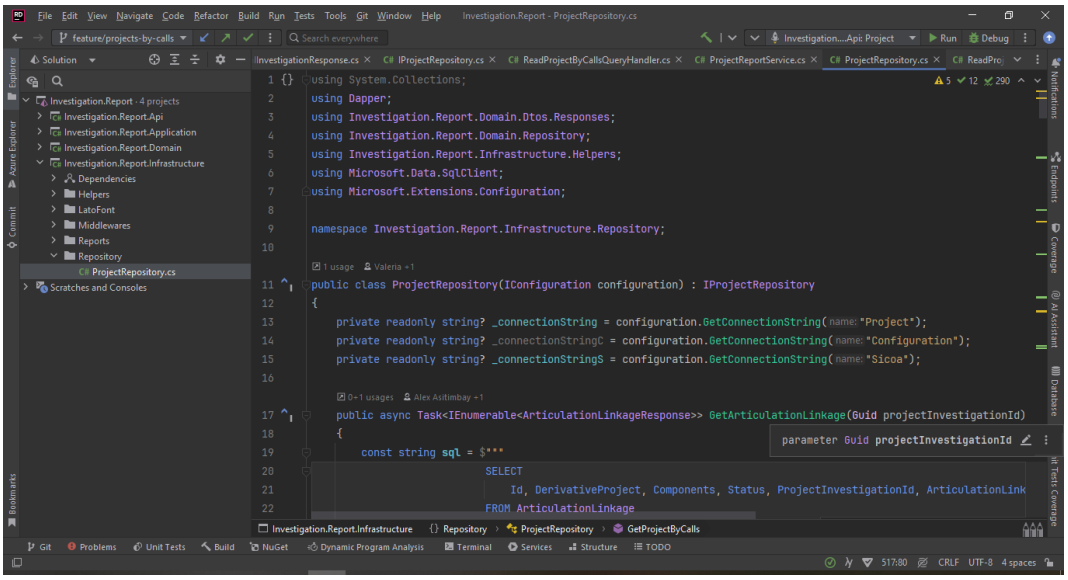


Figura 14: Capa de aplicación

En la capa Investigation.Report.Api, se puso en marcha un controlador que se comunica con el sistema mediante endpoints. Como se ilustra en la Figura 15, el método GetProjectCalls acepta una petición HTTP con el fin de producir un informe de los proyectos. En primer lugar, se comprobó el formato de visualización del informe para asegurarse de que estuviera en PDF o Excel; De lo contrario, se devuelve un error. Además, la petición contiene filtros concretos para recopilar únicamente la información relevante según los criterios establecidos.

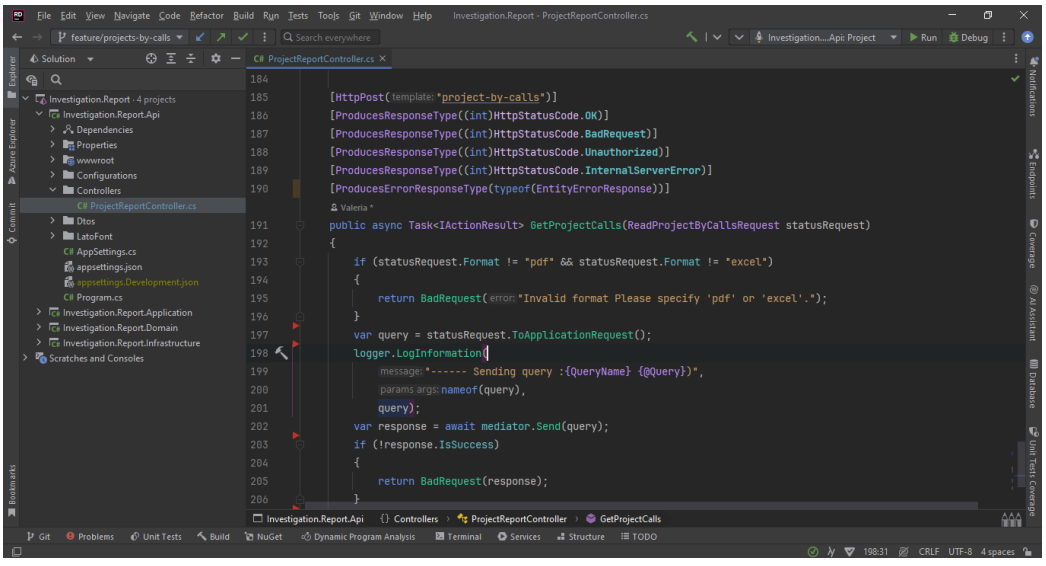


Figura 15: Controlador API

En la figura 16 se muestran los puntos de conexión que se crearon para la plantilla de informe del controller. La documentación de los endpoints es básica en la creación de APIs ya que proporciona la comprensión de cómo interactua con los diferentes servicios. Cuando se trata de API RESTful, herramientas como Swagger ayudan a crear documentación interactiva automáticamente, lo que facilita la exploración y prueba de puntos finales sin tener que completar solicitudes manuales de herramientas externas.

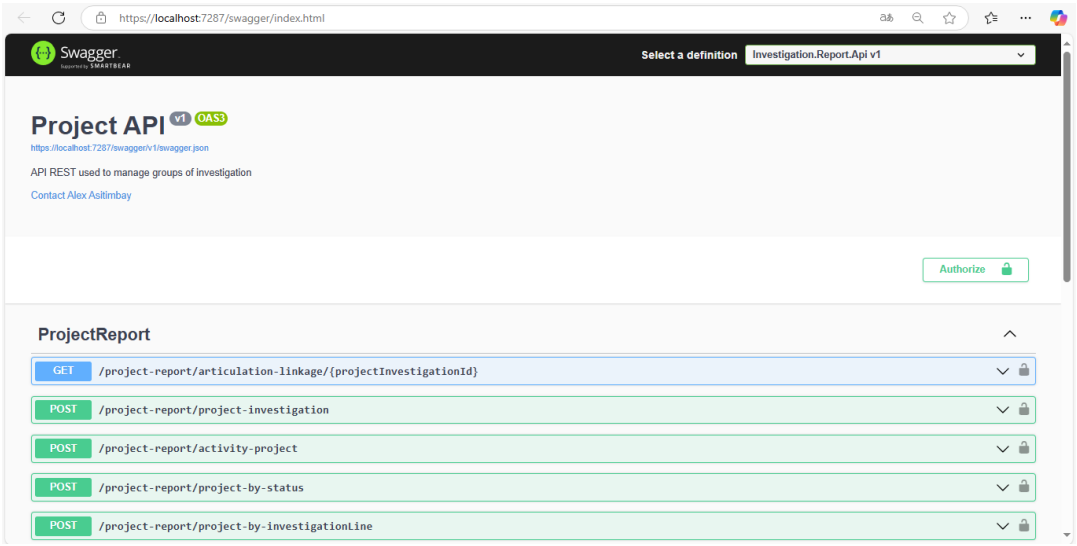


Figura 16: Endpoints

3.8.4 Lanzamiento

En la figura 17 se muestran los distintos roles disponibles en el sistema; esta vista brinda una gestión organizada de los roles, lo que proporciona a los usuarios la navegación y el camino a las funciones pertinentes y responsables centralmente del sistema.

Figura 17: Sistema de gestión de la investigación

Esta sección del sistema permite a los analistas generar informes basados en sus necesidades específicas, como se muestra en la Figura 18: Modelo de Reporteria para Proyectos de Investigación.

Figura 18: Módulo de reporteria

A continuación, en la Figura 19 se exponen los distintos filtros que se deben elegir para la visualización de un informe, y la mejor manera de describir cómo se diseñan se encuentra en el Anexo 1.

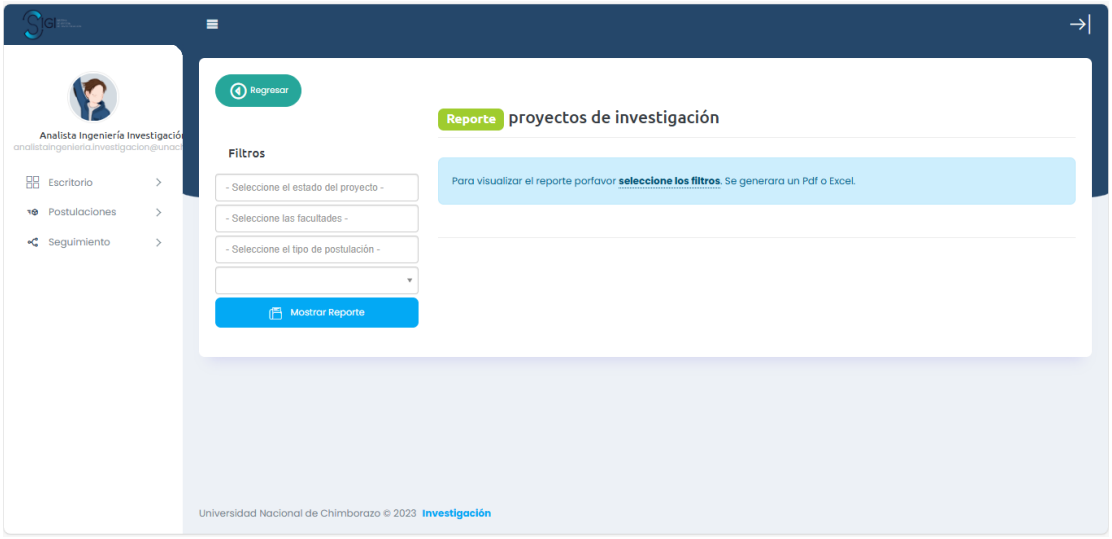


Figura 19: Filtros

En la Figura 20 se ilustra que el informe se visualizará antes de su entrega, haciendo hincapié en que está específicamente enfocado en los proyectos de investigación que se solicitan. Posibilitando a los analistas conseguir la información necesaria para obtener sus informes de los proyectos, suministrando así la toma de decisiones y el seguimiento adecuado de los mismos.

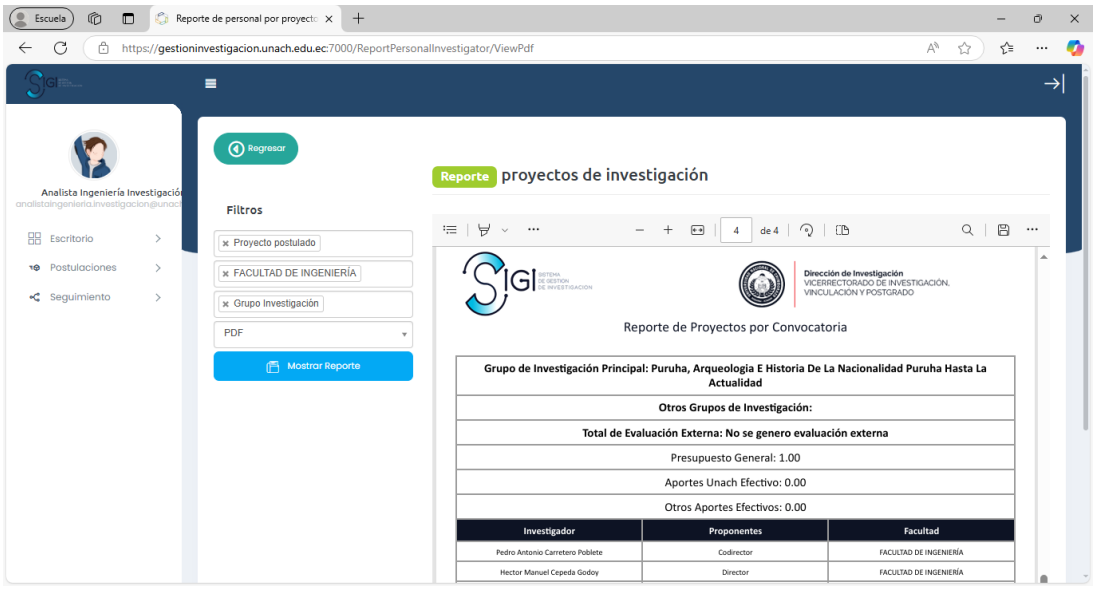


Figura 20: Reporte proyectos de investigación

3.8.5 Pruebas

Para cumplir con el objetivo 3 de valorar la eficacia del módulo de informes formado en el sistema de administración de investigación de la Universidad Nacional de Chimborazo, se realizaron diversas pruebas con el uso de Apache JMeter. Siguiendo los lineamientos establecidos por la norma ISO 25010, se ejecutaron 100 solicitudes con diversos propósitos en el transcurso de un segundo. Estas pruebas se realizaron en una computadora portátil. Luego, los datos recolectados fueron compilados y examinados, mostrando los resultados mediante diagramas estadísticos. El estudio se cotejo con las normas del modelo de calidad ISO 25010 para cada indicador, para confirmar la validez y confiabilidad de los hallazgos.

Parámetros de evaluación

Los indicadores son factores cruciales que se encargan de evaluar la eficacia del método; para ello, se utilizaron los siguientes indicadores, como se muestra en la Tabla 10.

Tabla 10. Indicadores

Medida	Indicadores
Eficiencia	Comportamiento Temporal Utilización de recursos • Uso de CPU • Uso de memoria RAM • Uso del disco Capacidad

Ejecución de las pruebas

En la Figura 21 se detallan los parámetros utilizados para configurar un escenario de prueba orientado a evaluar la calidad del módulo de reportes bajo los criterios establecidos por la norma ISO/IEC 25010. Esta norma internacional establece un modelo de calidad para los

productos de software y, en este caso, se centra en evaluar la eficiencia del rendimiento teniendo en cuenta factores como el comportamiento temporal, la utilización de recursos y la capacidad.

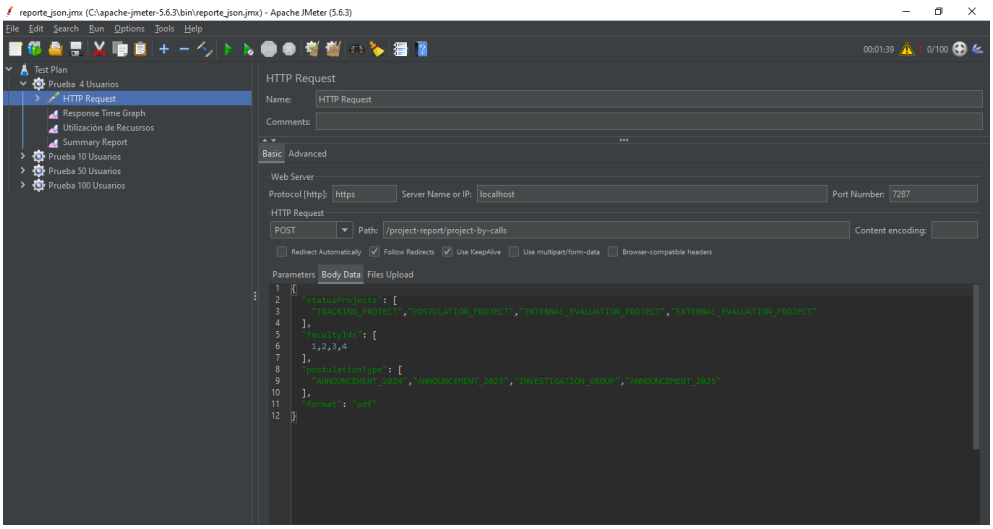


Figura 21: Escenario de pruebas

En En la figura 22, se creó una colección de hilos para simular 4,10,50 y 100 usuarios que remiten solicitudes en un periodo de 1 segundo. Estos valores se eligieron considerando que el módulo será utilizado únicamente por Analistas. Actualmente uno por cada facultad, sumando un total de 4, por lo que el número real de usuarios concurrentes es bajo. No obstante, se utilizaron escenarios con mayor cantidad de usuarios (hasta 100) con el objetivo de evaluar la capacidad del sistema bajo condiciones de estrés, más allá del uso habitual.

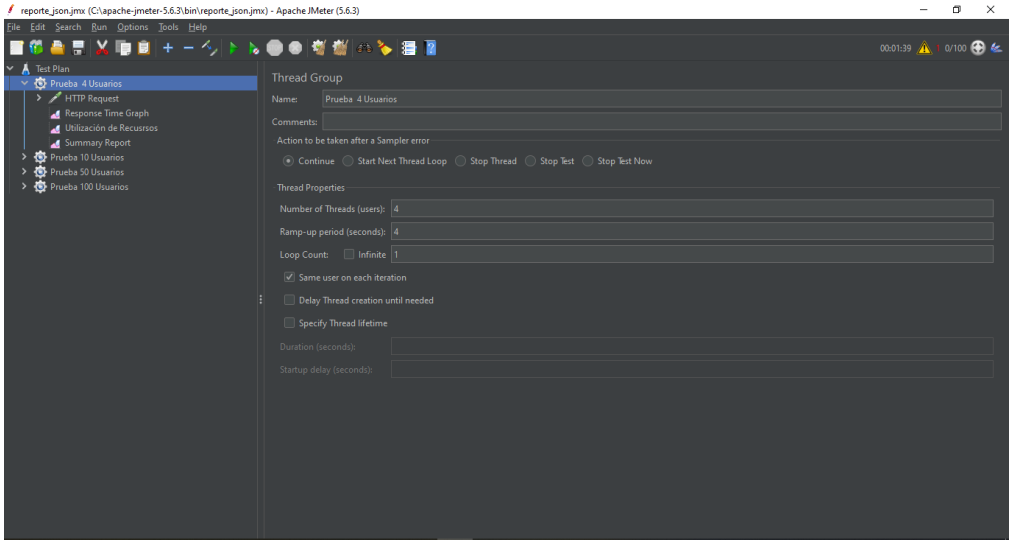


Figura 22: Configuración grupo de hilos

A continuación, la figura 23 ilustra cómo se utilizó el Informe Sumario para evaluar las métricas del comportamiento temporal y la idoneidad del módulo para varios usuarios.

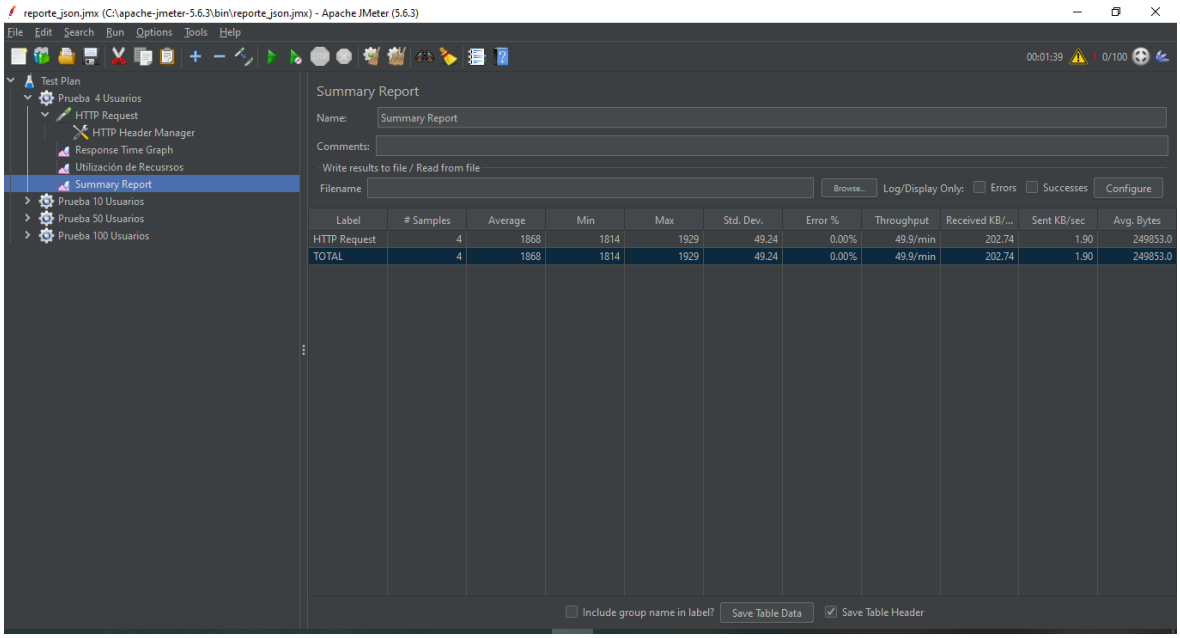


Figura 23: Resumen estadístico

En la Figura 24 se muestran los resultados de 100 usuarios simulados con un Gráfico de Tiempo de Respuesta, que se utiliza para visualizar los tiempos de respuesta de las solicitudes realizadas durante una prueba de carga.

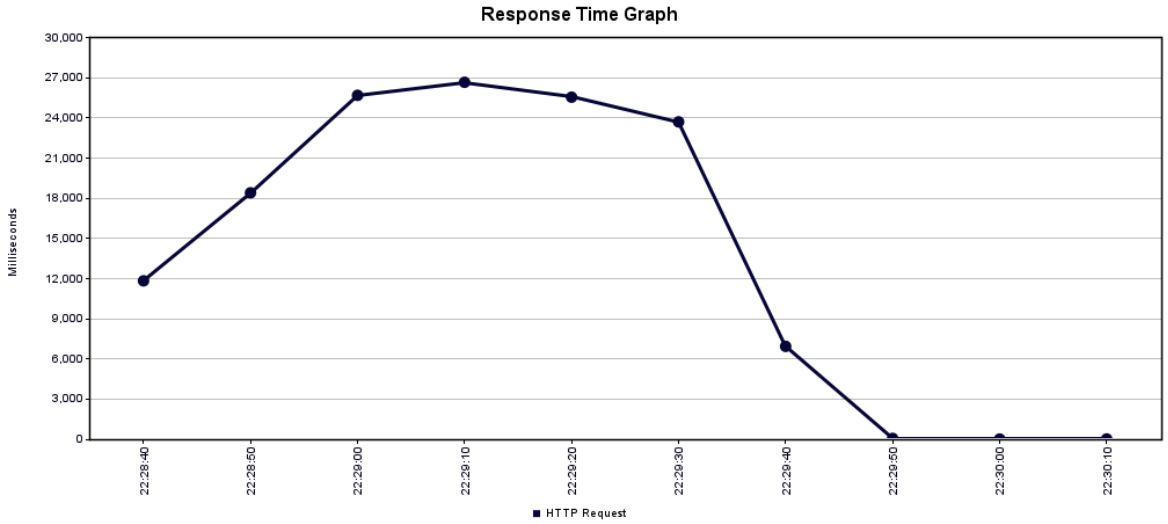


Figura 24: Carga Máxima

CAPÍTULO IV. RESULTADOS Y DISCUSIÓN

4.1 Interpretación de Resultados

Los hallazgos de las evaluaciones de eficacia realizadas se expusieron en el formato de reporte del Sistema de Gestión de la UNACH. JMeter fue el instrumento empleado para llevar a cabo estas pruebas (consulte el Anexo 2). Para evaluar la efectividad del método, se llevaron a cabo simulacros de varios niveles utilizando objetivos de evaluación. En los contextos de prueba de JMeter, se realizó una evaluación del estrés en el endpoint /project-report/project-by-calls, con un número variable de usuarios al mismo tiempo. Se registraron sus periodos de respuesta, fallos y la utilización de recursos para evaluar el desempeño del sistema en diferentes situaciones. Las pruebas realizadas para cada proceso se muestran en la Tabla 11, que también incluye 100 casos que fueron examinados.

Tabla 11. Métodos realizados

Procesos	Cantidad de pruebas
Prueba inicial con carga mínima de usuarios concurrentes.	4
Incremento moderado de carga simulando uso medio.	10
Prueba de carga alta con varios usuarios concurrentes.	50
Escenario de estrés con carga máxima esperada.	100

Cada prueba implicó el envío de solicitudes con los siguientes parámetros con formato JSON: statusProjects, facultyIds, postulationType y format. El punto de conexión se probó con cargas de 4, 10, 50 y 100 usuarios.

Los resultados de las pruebas realizadas se muestran en la figura 32 utilizando un gráfico al pastel.

Figura 25: Porcentaje

4.1.1 Valoración de indicadores

Con el fin de evaluar la efectividad del formato de informe implementado con Dapper, se establecieron métricas de evaluación basadas en ISO/IEC 25010. Esta norma proporciona un modelo de calidad de producto de software, en el que este estudio se centra en la eficiencia del rendimiento. Las principales métricas que se tienen en cuenta son:

- Comportamiento temporal: Evaluación tiempos de respuesta bajo diversos niveles de carga.
- Utilización de recursos: Análisis de los tipos y cantidades de recursos utilizados mientras el software realiza sus funciones
- Capacidad: Medición de la cantidad máxima de recursos que el sistema puede gestionar eficazmente cuando se realiza una acción concreta.

Este índice de desempeño se emplea para determinar si el sistema es capaz de gestionar un volumen considerable de peticiones sin sacrificar su capacidad de respuesta o estabilidad.

Comportamiento Temporal

Los resultados de las pruebas para 4, 10, 50 y 100 usuarios se resumen en la Tabla 12 y en la figura 26, que ayuda a analizar cómo cambiaron las métricas de rendimiento del módulo de informe a lo largo del tiempo.

Tabla 12. Resultados Comportamiento Temporal

Prueba	Tiempo Promedio (s)	Tiempo Mínimo (s)	Tiempo Máximo (s)
4	1.87	1.81	1.93
10	3.64	2.62	4.59
50	37.19	1.99	76.51
100	13.66	0.01	63.93

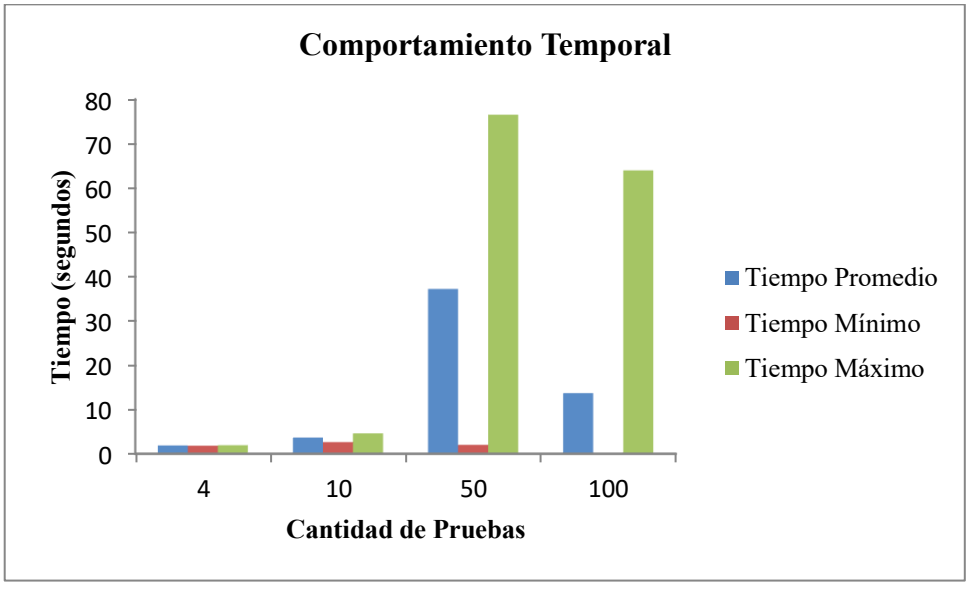


Figura 26: Resultado comportamiento temporal

Utilización de recursos

En la Tabla 13 y en la figura 27, se resumen los resultados de las pruebas realizadas con un número variable de usuarios, junto con el uso de la CPU, la RAM y el disco durante la ejecución de cada prueba. Estas métricas permiten observar el comportamiento del módulo bajo diversos niveles de demanda.

Tabla 13. Pruebas de Utilización de recursos

Prueba	Uso de CPU (%)	Uso de RAM (%)	Uso del Disco (%)
4	21	63-70	0
10	21- 28	63-70	7-14
50	21-28	63-70	7-14
100	35-42	63-70	7-14

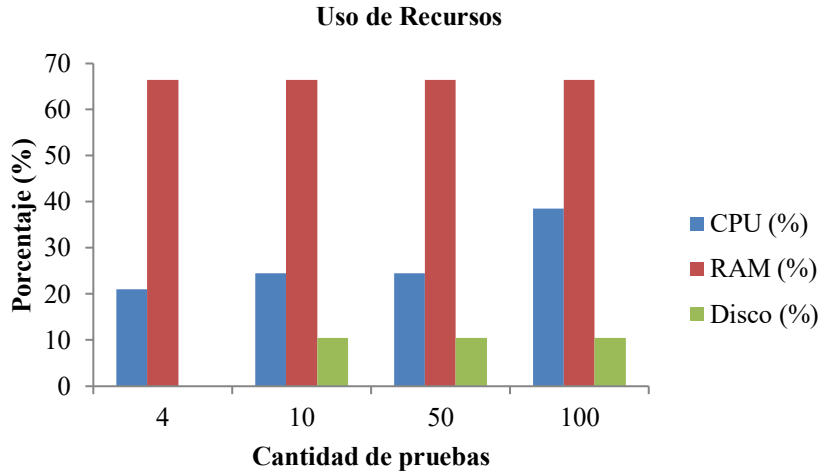


Figura 27: Resultado utilización de recursos

Capacidad

En la tabla 14 y en la Figura 28 se muestran los resultados obtenidos teniendo en cuenta el bajo rendimiento de varias cargas de usuarios que compiten entre sí. En esta prueba se utilizó el indicador Throughput (solicitudes/segundo), que indica el número promedio de solicitudes que el sistema puede procesar en un segundo.

Tabla 14. Prueba de Capacidad

Prueba	Rendimiento (req/s)	Usuarios Concurrentes	Errores (%)
4	0.831	4	0.0
10	0.845	10	0.0
50	0.474	50	0.0
100	1.010	100	0.4

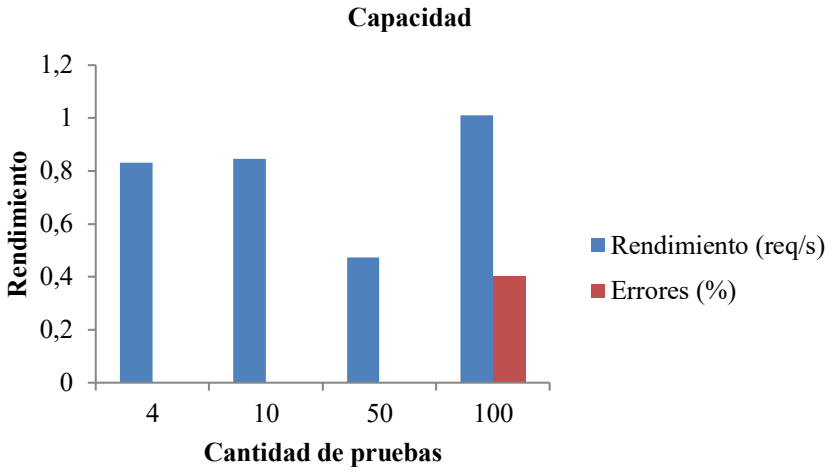


Figura 28: Resultado de capacidad

Comparación de los valores obtenidos con los resultados de eficiencia de desempeño.

La Tabla 15 y la Figura 29 sintetizan la comparación entre los resultados logrados en los análisis y los criterios de aceptación en estos exámenes. Es importante destacar que los criterios de aceptación se establecieron ya que el estándar de eficiencia de rendimiento carece de criterios establecidos. Este procedimiento ofrece datos acerca de los elementos que el sistema sobrepasa, en comparación con los estándares fijados, y de los que requieren mejora en el sistema de reportes.

Tabla 15. Comparación de los valores obtenidos

Indicador	Criterio de aceptación	Resultado	Cumplimiento
Comportamiento temporal	Tiempo de Respuesta <= 15 seg	14.84s	100%
Utilización de Recursos	CPU <= 50%, RAM <= 70%, Disco <= 10%	CPU 31.25%, RAM 63.25%, Disco 6.75%	CPU: 100%, RAM: 100%, Disco: 75%
Capacidad	Throughput ≥ 1.0 req/s	0.83 req/s	83%

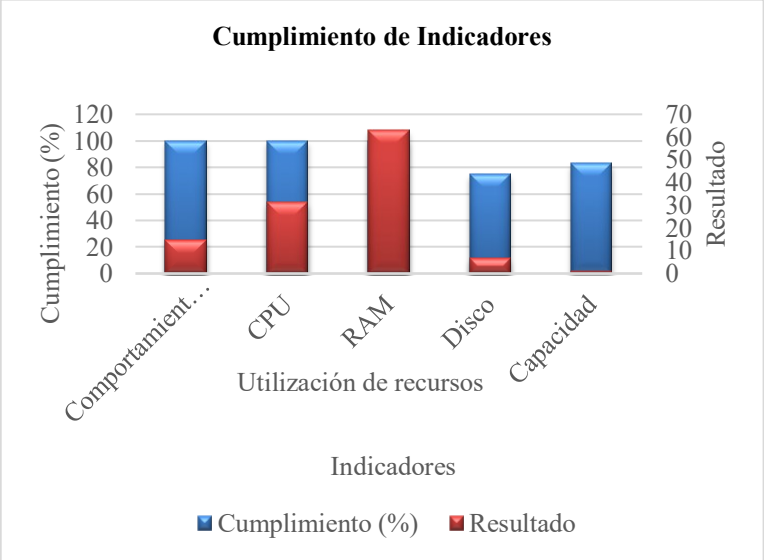


Figura 29: Comparación de los resultados

4.2 Discusión

A partir de los hallazgos, se puede concluir que el rendimiento del módulo de informes ha mejorado desde que Dapper se integró en el Sistema de Gestión de Investigación. El comportamiento temporal cumple con el 100% de los requisitos hasta 15 segundos, lo cual es un resultado completamente satisfactorio. Con respecto al uso de recursos, el sistema demostró eficiencia con una utilización promedio de CPU del 31.25%, el uso de RAM fluctuando entre el 63% y el 70%, y el uso de disco en el 6.75%, resultando en un 91.6% de cumplimiento para este indicador. Por otro lado, en lo que respecta al volumen de solicitudes procesadas, el sistema logró un cumplimiento del 83 por ciento a una tasa de 0.83 solicitudes por segundo. En general, las características más destacadas de Dapper son sus contribuciones positivas hacia el desarrollo de las métricas de efectividad de las normas ISO/IEC 25010.

Positivas son las conclusiones en términos de rendimiento luego de la integración de Dapper en el sistema de reporting dentro del Sistema de Gestión de Investigación, en comparación con el estudio realizado por [11], en el cual se analizó el desempeño de Dapper y Entity Framework. En ese estudio en particular, se afirmó que Dapper tenía mejores tiempos de respuesta mientras que Entity Framework tenía una ventaja significativa en la utilización de recursos y capacidad; el estudio actual muestra que Dapper proporciona un equilibrio estable entre velocidad y eficiencia. La metodología implementada afectó el comportamiento de Dapper bajo diferentes cargas de forma estable, con control de los recursos del sistema, sin errores en la prueba de reporte de Dapper, lo que confirma que Dapper es una solución efectiva para ambientes de rendimiento crítico, en especial para el procesamiento de reportes masivos

CAPÍTULO V. CONCLUSIONES y RECOMENDACIONES

5.1 Conclusiones

Efectuar un análisis sistemático de la bibliografía acerca de las tecnologías y normas pertinentes para la integración de Dapper con servicios API REST proporciona una base teórica vigente. Este estudio facilita la identificación del arte contemporáneo, en este escenario, el "estado del arte", que se compone de las mejores prácticas y las carencias asociadas al empleo de estas tecnologías en su totalidad, lo que facilita una aplicación más avanzada en el futuro.

Éxito se logró alcanzar en la implementación del Dapper para generación de reportes en el Sistema de Gestión de Investigación, por cuanto mejora el servicio y la agilidad con los datos REST en la API, pues interactúa con el sistema. Esto no solo optimiza el tiempo de respuesta a las solicitudes, sino que mejora, el tiempo en que se procesan los reportes, la satisfacción general del usuario.

La evaluación sobre la efectividad del formato del informe, que fue realizada por el estándar ISO/IEC 25010, arrojó resultados positivos. El formato cumplió con los indicadores establecidos respecto al comportamiento temporal, utilización de recursos y capacidad. A pesar de que existieron algunos aspectos donde el comportamiento temporal del sistema pudiera ser mejorado, en líneas generales, el módulo tuvo un desempeño adecuado especialmente en el economizador de recursos a nivel de sistema: CPU, RAM, disco, además, el resultado barrido cumplió con criterios de aceptación en la mayoría de los casos.

5.2 Recomendaciones

Investigar nuevas tecnologías que aporten el uso de Dapper en .NET mediante API REST, esto ayudará a fortalecer la arquitectura del sistema.

Con el objetivo principal de mejorar el sistema en lo que respecta al tiempo de respuesta y reducir la carga en los recursos del sistema, realice revisiones periódicas de las consultas SQL implementadas dentro del módulo.

Es recomendable explorar nuevas técnicas que ayuden a mejorar el desempeño, particularmente cuando el sistema opera con un alto volumen de usuarios; esto puede incluir esfuerzos para mejorar el desempeño de las sentencias SQL.

BIBLIOGRÁFIA

[1] Universidad Nacional de Chimborazo, «Gestión de la Investigación,» https://www.unach.edu.ec/gestion-de-la-investigacion-ele/.

[2] Y. Khalid, «ScrapingBee,» 30 1 2023. https://www.scrapingbee.com/blog/six-characteristics-of-rest-api/.

[3] MuleSoft, «MuleSoft,» https://www.mulesoft.com/api/rest/top-3-benefits-of-rest-apis#:~:text=1.-,Lightweight,of%20things%20devices%2C%20and%20more..

[4] admin, «¿Qué es Dapper?,» https://render2web.com/net-6/dapper/que-es-dapper/.

[5] «Dapper queries synchronized with MSSQL database schema · by Alex Code blog,» Byalexblog.net.

[6] F. Parikh. https://www.csharp.com/UploadFile/e4e3f7/dapper-king-of-micro-orm-C-Sharp-net/.

[7] Proyectos ZZZ, «Learndapper,» 17 10 2024. https://www.learndapper.com/.

[8] S. M. A. Khan, «Csharp,» ©2025 CSharp.com., https://www.csharp.com/article/api-development-using-dapper-and-microsoft-asp-net-core-web-api/.

[9] kexugit, «Microsoft.com,» https://learn.microsoft.com/es-es/archive/msdn-magazine/2016/may/data-points-dapper-entity-framework-and-hybrid-apps?form=MG0AV3.

[10] A. X. Minango Guatumillo, J. O. Silva Rivera y P. A. Buñay Guisñan, «Análisis del Desempeño entre Dapper y Entity Framework, caso Aplicativo: Sistema de Buenas Prácticas en Empresas Turísticas de Riobamba,» Unach.edu.ec, http://dspace.unach.edu.ec/handle/51000/5493.

[11] A. . E. GÜVERCİN y B. AVENOGLU, «Bilişim Teknolojileri Dergisi,» 2022. https://dergipark.org.tr/en/pub/gazibtd/issue/73288/1059516.

[12] JetBrains, «JetBrains,» [En línea]. Available: https://www.jetbrains.com/rider/.

[13] rwestMSFT, «SQL Server Management Studio (SSMS),» https://learn.microsoft.com/es-es/ssms/sql-server-management-studio-ssms?view=sql-server-ver16.

[14] «Quick start | QuestPDF,» https://www.questpdf.com/quick-start.html.

[15] W. contributors, «Office Open XML,» 15 9 2024. https://en.wikipedia.org/wiki/Office_Open_XML?form=MG0AV3.

[16] Manzanelli, «Norma ISO 25010,» NormasISO.org, 24 9 2023. https://normasiso.org/norma-iso-25010/?form=MG0AV3. [Último acceso: 29 10 2024].

[17] B. Paneiva, «Checklist ISO 25010 de Requisitos de Calidad de Sistemas y Software,» Lumiform, 10 11 2023. https://lumiformapp.com/es/checklists-recursos/certificacion-iso-25010?form=MG0AV3. [Último acceso: 29 10 2024].

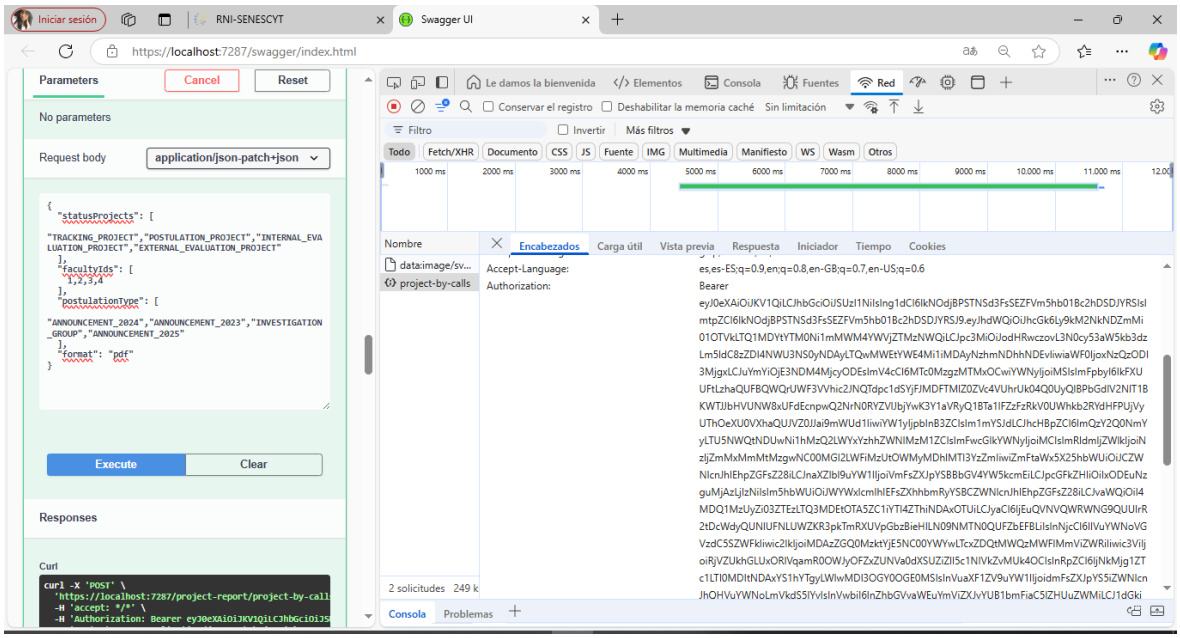
[18] «ISO 25010,» 2024. https://iso25000.com/index.php/normas-iso-25000/iso-25010.

[19] J. Martins, «¿Qué es la metodología Kanban y cómo funciona?,» https://asana.com/es/resources/what-is-kanban.

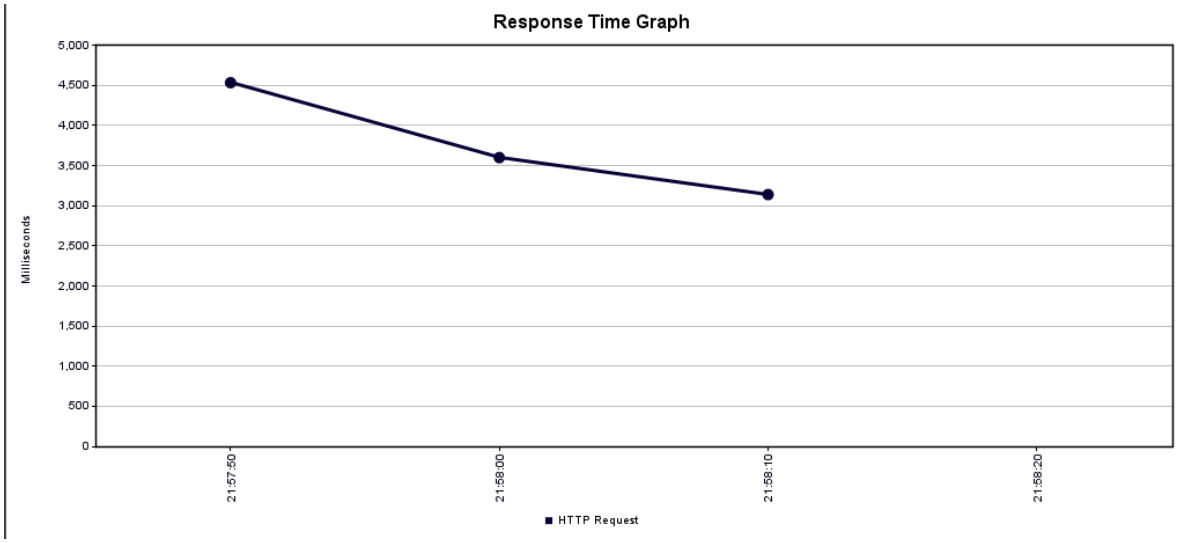
[20] I. Lara, «Tecnocrática,» 25 9 2023. https://tecnocratica.net/que-es-api-rest-para-que-sirve-ejemplos/#:~:text=Una%20API%20REST%20(Representational%20State,de%20datos%20entre%20diferentes%20sistemas..

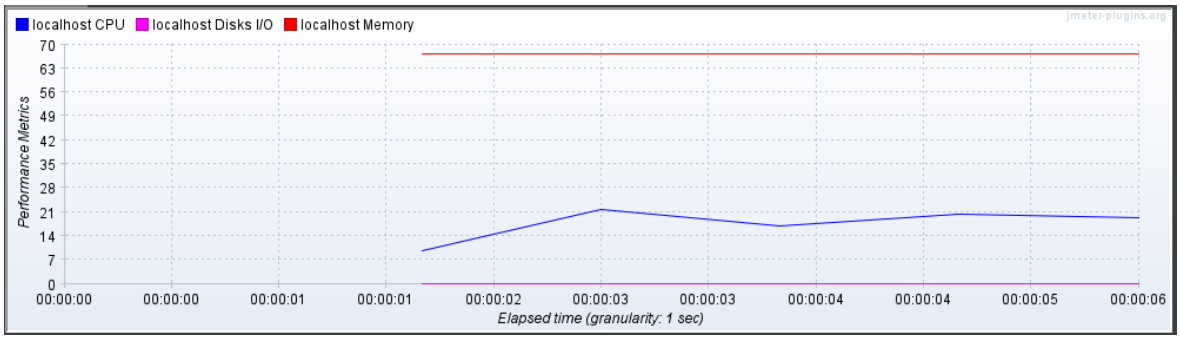
ANEXOS

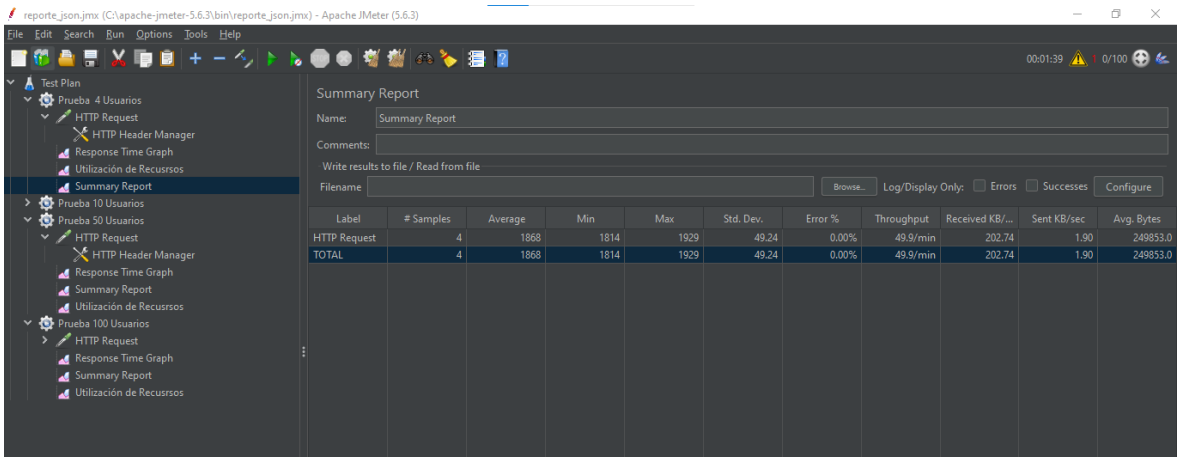
Anexo 1

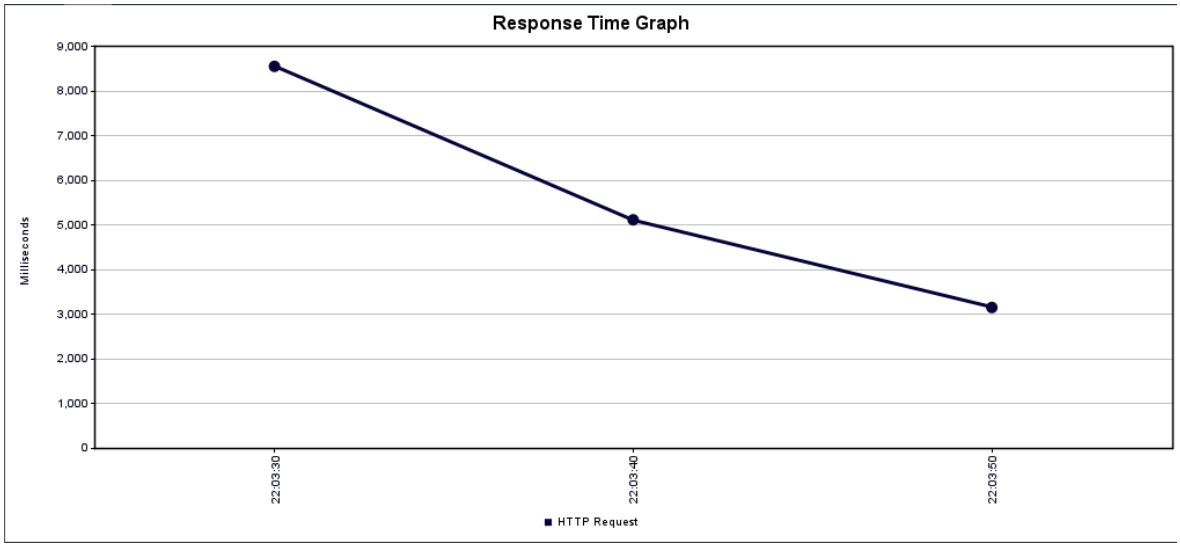


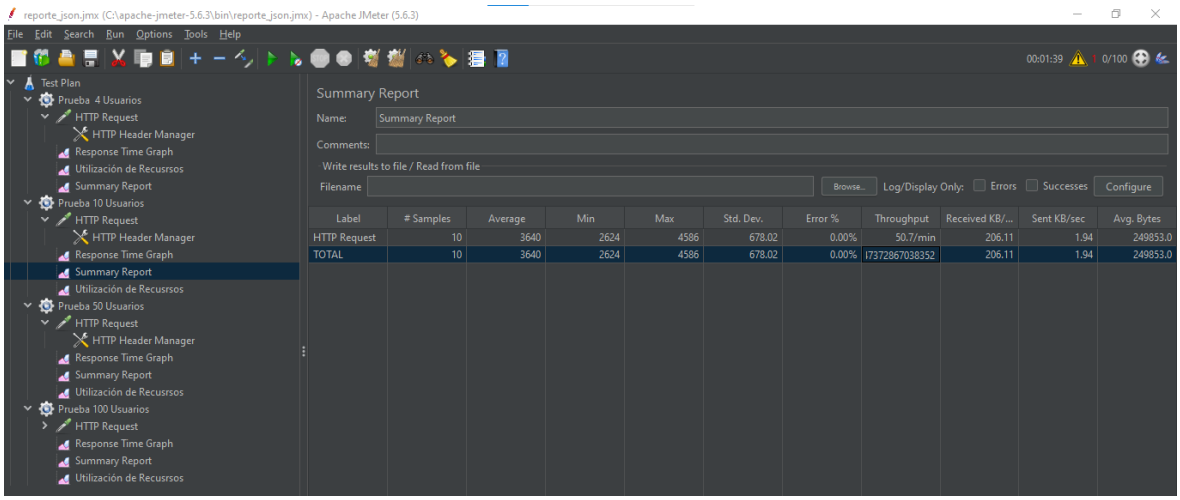
Anexo 2

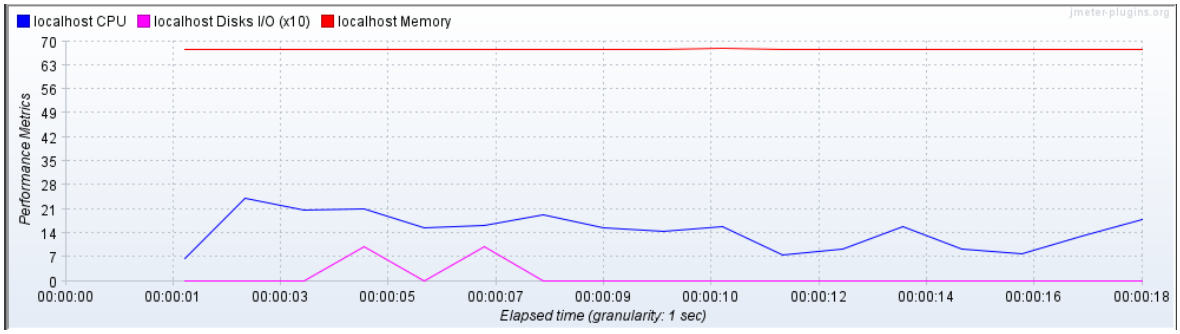


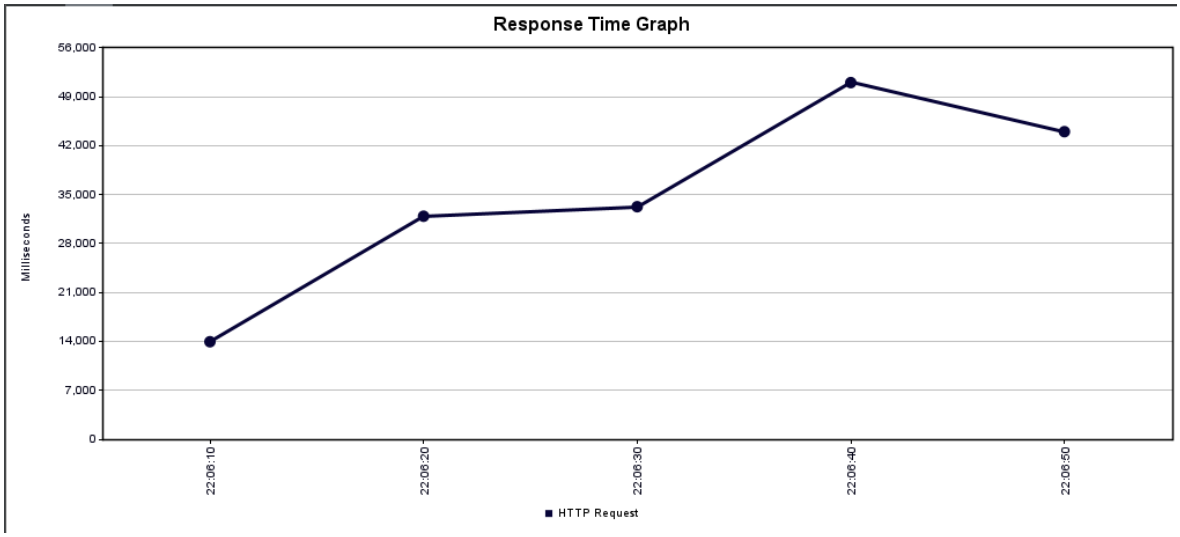

localhost CPU
localhost Disks I/O
localhost Memory
jmeter-plugins.org
Performance Metrics
70
63
56
49
42
35
28
21
14
7
0
00:00:00
00:00:00
00:00:01
00:00:01
00:00:02
00:00:03
00:00:03
00:00:04
00:00:04
00:00:05
00:00:06
Elapsed time (granularity: 1 sec)

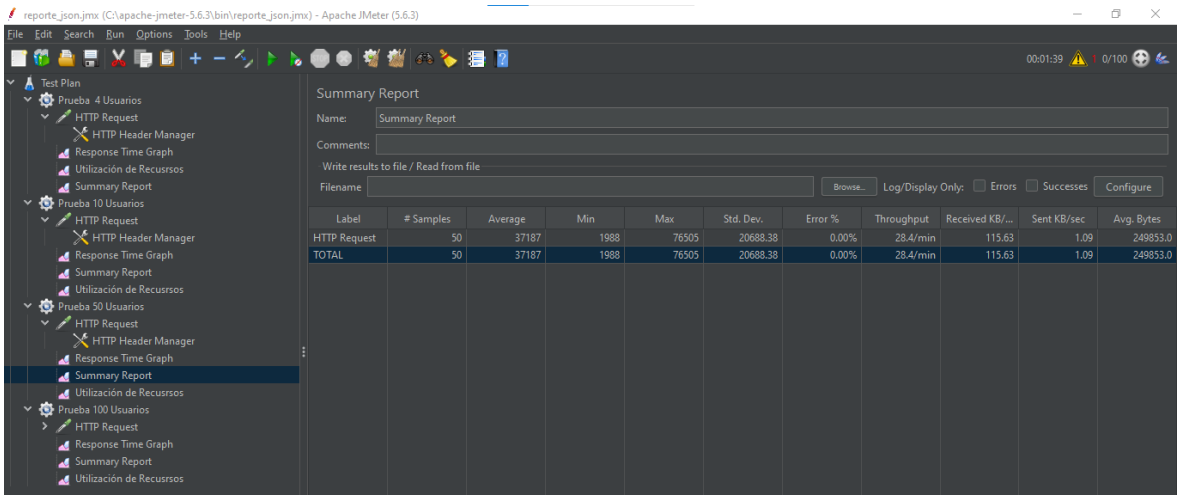

reporte_json.jmx (C:\apache-jmeter-5.6.3\bin\reporte_json.jmx) - Apache JMeter (5.6.3)
File Edit Search Run Options Tools Help
00:01:39 0/100
Test Plan
Prueba 4 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Utilización de Recursos
Summary Report
Prueba 10 Usuarios
Prueba 50 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Summary Report
Utilización de Recursos
Prueba 100 Usuarios
HTTP Request
Response Time Graph
Summary Report
Utilización de Recursos
Summary Report
Name: Summary Report
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only: Errors Successes
Configure
Label # Samples Average Min Max Std. Dev. Error % Throughput Received KB/... Sent KB/sec Avg. Bytes
HTTP Request 4 1868 1814 1929 49.24 0.00% 49.9/min 202.74 1.90 249853.0
TOTAL 4 1868 1814 1929 49.24 0.00% 49.9/min 202.74 1.90 249853.0

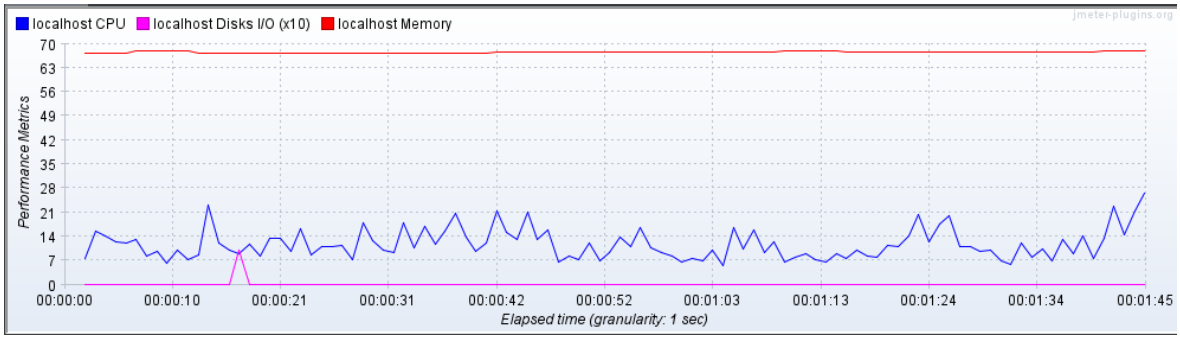

Response Time Graph
Milliseconds
9,000
8,000
7,000
6,000
5,000
4,000
3,000
2,000
1,000
0
22:03:30
22:03:40
22:03:50
HTTP Request

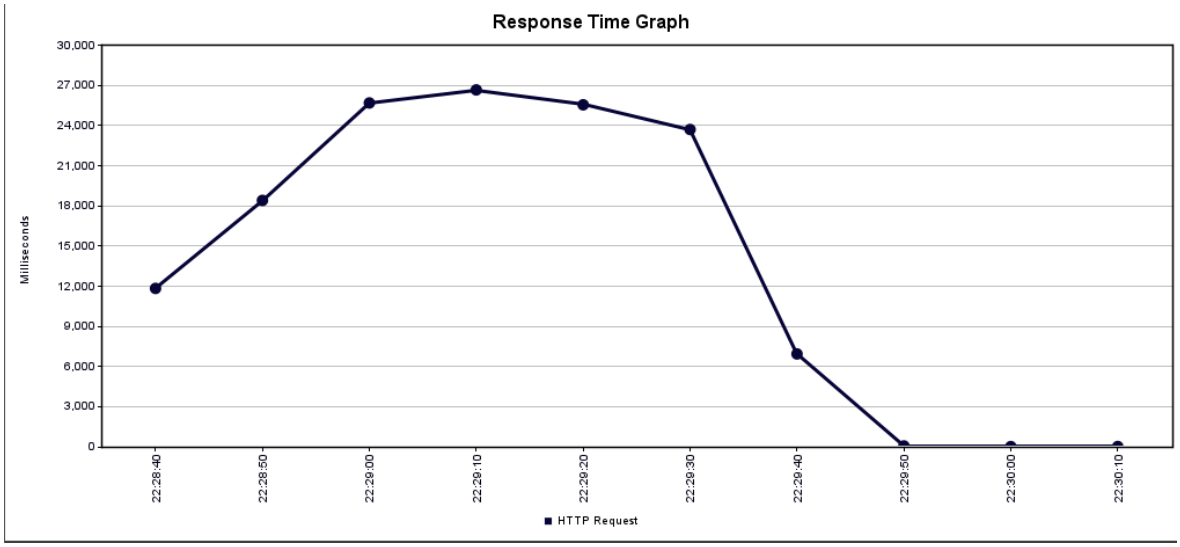

reporte_json.jmx (C:\apache-jmeter-5.6.3\bin\reporte_json.jmx) - Apache JMeter (5.6.3)
File Edit Search Run Options Tools Help
Test Plan
Prueba 4 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Utilización de Recursos
Summary Report
Prueba 10 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Summary Report
Utilización de Recursos
Prueba 50 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Summary Report
Utilización de Recursos
Prueba 100 Usuarios
HTTP Request
Response Time Graph
Summary Report
Utilización de Recursos
Summary Report
Name: Summary Report
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Label # Samples Average Min Max Std. Dev. Error % Throughput Received KB/... Sent KB/sec Avg. Bytes
HTTP Request 10 3640 2624 4586 678.02 0.00% 50.7/min 206.11 1.94 249853.0
TOTAL 10 3640 2624 4586 678.02 0.00% 17372867038352 206.11 1.94 249853.0

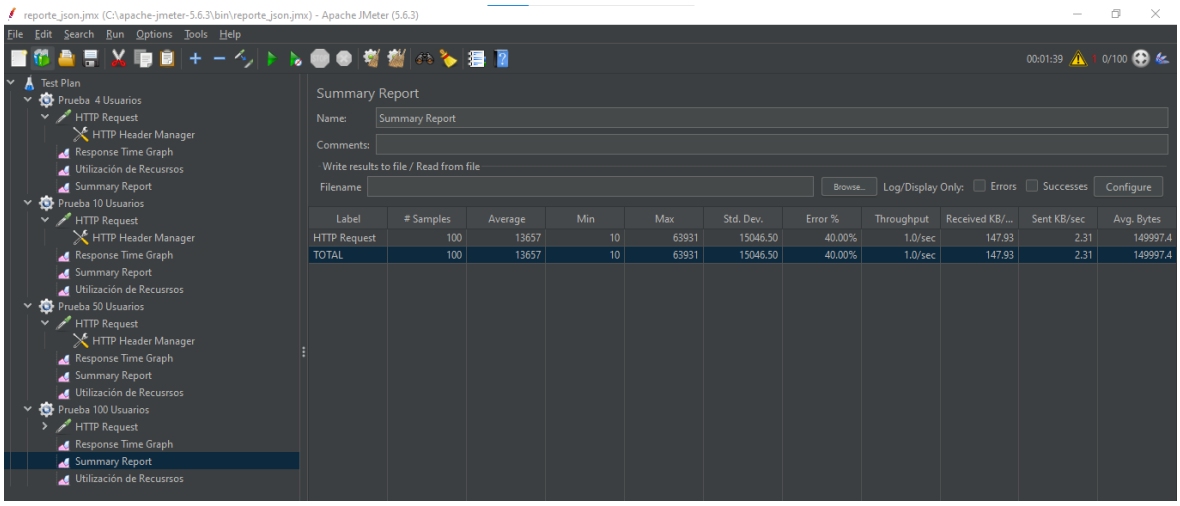

localhost CPU
localhost Disks I/O (x10)
localhost Memory
jmeter-plugins.org
Performance Metrics
70
63
56
49
42
35
28
21
14
7
0
00:00:00
00:00:01
00:00:03
00:00:05
00:00:07
00:00:09
00:00:10
00:00:12
00:00:14
00:00:16
00:00:18
Elapsed time (granularity: 1 sec)


Response Time Graph
Milliseconds
56,000
49,000
42,000
35,000
28,000
21,000
14,000
7,000
0
22:06:10
22:06:20
22:06:30
22:06:40
22:06:50
HTTP Request


reporte_json.jmx (C:\apache-jmeter-5.6.3\bin\reporte_json.jmx) - Apache JMeter (5.6.3)
File Edit Search Run Options Tools Help
Test Plan
Prueba 4 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Utilización de Recursos
Summary Report
Prueba 10 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Summary Report
Utilización de Recursos
Prueba 50 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Summary Report
Utilización de Recursos
Prueba 100 Usuarios
HTTP Request
Response Time Graph
Summary Report
Utilización de Recursos
Summary Report
Name: Summary Report
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Label # Samples Average Min Max Std. Dev. Error % Throughput Received KB/... Sent KB/sec Avg. Bytes
HTTP Request 50 37187 1988 76505 20688.38 0.00% 28.4/min 115.63 1.09 249853.0
TOTAL 50 37187 1988 76505 20688.38 0.00% 28.4/min 115.63 1.09 249853.0


localhost CPU
localhost Disks I/O (x10)
localhost Memory
jmeter-plugins.org
Performance Metrics
70
63
56
49
42
35
28
21
14
7
0
00:00:00
00:00:10
00:00:21
00:00:31
00:00:42
00:00:52
00:01:03
00:01:13
00:01:24
00:01:34
00:01:45
Elapsed time (granularity: 1 sec)


Response Time Graph
Milliseconds
30,000
27,000
24,000
21,000
18,000
15,000
12,000
9,000
6,000
3,000
0
22:28:40
22:28:50
22:29:00
22:29:10
22:29:20
22:29:30
22:29:40
22:29:50
22:30:00
22:30:10
HTTP Request


reporte_json.jmx (C:\apache-jmeter-5.6.3\bin\reporte_json.jmx) - Apache JMeter (5.6.3)
File Edit Search Run Options Tools Help
00:01:39 0/100
Test Plan
Prueba 4 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Utilización de Recursos
Summary Report
Prueba 10 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Summary Report
Utilización de Recursos
Prueba 50 Usuarios
HTTP Request
HTTP Header Manager
Response Time Graph
Summary Report
Utilización de Recursos
Prueba 100 Usuarios
HTTP Request
Response Time Graph
Summary Report
Utilización de Recursos
Summary Report
Name: Summary Report
Comments:
Write results to file / Read from file
Filename
Browse...
Log/Display Only:
Errors
Successes
Configure
Label # Samples Average Min Max Std. Dev. Error % Throughput Received KB/... Sent KB/sec Avg. Bytes
HTTP Request 100 13657 10 63931 15046.50 40.00% 1.0/sec 147.93 2.31 149997.4
TOTAL 100 13657 10 63931 15046.50 40.00% 1.0/sec 147.93 2.31 149997.4