JINTIKOM
JURNAL INFORMASI TEKNOLOGI DAN KOMPUTER
https://journal.umbogorraya.ac.id/index.php/jintikom

Peningkatan Performa Aplikasi Web Dinamis Berbasis PHP melalui Implementasi Redis Caching

Suwarjono[1], Fitrah Averoes[2]
[1,2] Departemen Ilmu Komputer, Fakultas Kesehatan dan Sains
Universitas Muhammadiyah Bogor Raya
Jl. Raya Leuwiliang No.106, Kec. Leuwiliang, Kabupaten Bogor, Jawa Barat.

Volume 1 Nomor 1
Maret 2025: 37-44

Article History
Submission: 10-02-2025
Revised: 22-02-2025
Accepted: 21-03-2025
Published: 28-03-2025

Kata Kunci:
Redis, caching, PHP, MySQL, performa web, in-memory, optimasi aplikasi

Keywords:
Redis, caching, PHP, MySQL, web performance, in-memory, application optimization

Korespondensi:
(Suwarjono)
(Telp.)
(bsuwarjono@gmail.com)

Abstrak: Performa aplikasi web merupakan faktor krusial dalam meningkatkan kepuasan dan keterlibatan pengguna. Dalam pengembangan aplikasi PHP berbasis database, beban query yang tinggi menjadi tantangan yang menghambat waktu respon sistem. Penelitian ini mengkaji penggunaan Redis sebagai teknologi caching in-memory untuk mengurangi latensi dan beban query MySQL pada aplikasi PHP sederhana. Eksperimen dilakukan dalam lingkungan XAMPP dengan dua skenario pengujian: tanpa Redis dan dengan Redis caching. Pengukuran dilakukan menggunakan Apache Benchmark dan logger internal untuk mencatat waktu respon dan jumlah query aktual. Hasil menunjukkan Redis mampu menurunkan waktu respon dari 412 ms menjadi 96 ms dan menurunkan jumlah query hingga 83%. Studi ini memperkuat temuan sebelumnya mengenai efektivitas Redis dalam sistem web dinamis dan memberikan panduan teknis implementasi caching dalam konteks lingkungan lokal.

***Abstract**: The performance of web applications is a crucial factor in enhancing user satisfaction and engagement. In the development of database-driven PHP applications, high query loads pose a challenge that hampers system response times. This study investigates the use of Redis as an in-memory caching technology to reduce latency and MySQL query load in a simple PHP application. Experiments were conducted in a XAMPP environment under two testing scenarios: without Redis and with Redis caching. Measurements were carried out using Apache Benchmark and an internal logger to record response times and the actual number of queries. The results demonstrate that Redis was able to reduce response time from 412 ms to 96 ms and decrease the number of queries by up to 83%. This study reinforces previous findings regarding the effectiveness of Redis in dynamic web systems and provides technical guidance for implementing caching in a local development environment.*

PENDAHULUAN

Performa aplikasi web merupakan aspek fundamental dalam pengembangan sistem digital modern karena berperan langsung dalam membentuk pengalaman pengguna. Seiring meningkatnya ketergantungan masyarakat terhadap layanan daring—seperti sistem informasi akademik, e-commerce, perbankan digital, dan layanan publik berbasis web—ekspektasi terhadap kecepatan akses dan responsivitas sistem juga meningkat tajam. Penelitian oleh Google menyebutkan bahwa keterlambatan waktu muat lebih dari tiga detik dapat menyebabkan lebih dari 50% pengguna meninggalkan situs, menunjukkan bahwa waktu respons menjadi indikator vital dalam retensi pengguna (Somerville, 2016).

Dalam praktiknya, banyak aplikasi web dikembangkan menggunakan PHP sebagai bahasa pemrograman sisi server yang populer dan relatif mudah dipelajari. PHP memiliki komunitas besar dan ekosistem pendukung yang luas, serta kompatibel dengan berbagai sistem basis data seperti MySQL dan PostgreSQL. Oleh karena itu, PHP masih menjadi pilihan utama dalam pengembangan web dinamis, termasuk di lingkungan pendidikan, UMKM, dan startup (Li, Jiang, & Shi, 2017).

Sebagai sarana pengembangan lokal, banyak pengembang pemula maupun profesional memanfaatkan XAMPP, yaitu paket perangkat lunak yang menyediakan Apache (web server), MySQL/MariaDB (basis data), PHP, dan Perl dalam satu paket yang mudah diinstal dan dijalankan di sistem operasi Windows, Linux, dan macOS. Lingkungan XAMPP memungkinkan simulasi dan pengujian aplikasi web secara lokal sebelum diterapkan ke server produksi, menjadikannya alat vital dalam proses pengembangan iteratif (Vishwasrao, 2017).

Namun demikian, tantangan kinerja sering muncul ketika aplikasi mengalami peningkatan lalu lintas atau permintaan data yang berulang. Setiap permintaan ke server yang memerlukan query ke basis data dapat memperlambat sistem jika tidak ditangani secara efisien. Beban berulang inilah yang menjadi akar dari latency tinggi, terlebih jika struktur query tidak

optimal atau basis data tumbuh besar seiring waktu.

Caching hadir sebagai salah satu solusi efektif untuk mengatasi persoalan ini. Dengan menyimpan hasil query atau data statis ke dalam penyimpanan sementara berkecepatan tinggi (biasanya memori), sistem dapat menghindari pemrosesan ulang untuk permintaan yang sama. Salah satu teknologi caching yang saat ini banyak digunakan adalah Redis, yaitu penyimpanan key-value in-memory yang mendukung berbagai struktur data seperti string, hash, dan list. Redis terkenal dengan kecepatan luar biasa dan latensinya yang sangat rendah, sehingga cocok untuk aplikasi dengan kebutuhan akses cepat dan frekuensi tinggi (George, 2021; Chen et al., 2016).

Berbagai studi telah menunjukkan bahwa penggunaan Redis mampu menurunkan waktu respons aplikasi web secara signifikan. Misalnya, Kusuma et al. (2017) membandingkan berbagai metode caching dan menemukan bahwa Redis lebih unggul dalam efisiensi dan konsistensi data. Selain itu, Gill dan Goyal (2021) menunjukkan bahwa Redis dapat meningkatkan throughput sistem berbasis Laravel hingga empat kali lipat.

Walaupun demikian, implementasi Redis memerlukan perhatian khusus, terutama dalam pengaturan Time To Live (TTL), invalidasi cache, dan pemantauan penggunaan memori. Jika tidak dikonfigurasi dengan baik, Redis dapat menyimpan data lama (stale) yang tidak mencerminkan perubahan terkini di database, atau justru menimbulkan bottleneck baru karena pemanfaatan RAM yang berlebihan (Raghav, 2018).

Dengan latar belakang tersebut, penelitian ini bertujuan untuk mengevaluasi secara eksperimental pengaruh Redis terhadap kinerja aplikasi PHP dalam lingkungan pengembangan lokal berbasis XAMPP. Fokus diberikan pada dua indikator utama, yaitu waktu respons sistem dan jumlah query SQL yang terjadi, sebagai ukuran efisiensi caching. Penelitian ini juga menyajikan pendekatan teknis dan praktis yang relevan untuk pengembang pemula hingga menengah dalam

mengintegrasikan Redis dalam proses pengembangan aplikasi berbasis PHP.

METODE

1. Desain Eksperimen

Penelitian ini mengadopsi pendekatan kuantitatif eksperimental dengan pengujian dua skenario sistem: (1) tanpa Redis dan (2) dengan Redis. Setiap skenario dijalankan sebanyak 30 kali untuk menjamin validitas data (Creswell, 2014).

2. Lingkungan Pengujian

- **Perangkat keras**: PC dengan prosesor AMD Ryzen 5, RAM 8 GB
- **Sistem Operasi**: Windows 10, Redis v5.0 diakses melalui WSL
- **Perangkat lunak**: XAMPP v7.4.33, PHP 7.4, MySQL 5.7, dan pustaka Predis untuk koneksi Redis

3. Implementasi Redis

Caching diterapkan pada endpoint PHP yang menghasilkan data dari query MySQL. Sistem memeriksa cache sebelum mengeksekusi query. Jika data tersedia, ia langsung disajikan; jika tidak, hasil query disimpan di Redis dengan TTL (Time To Live) selama 5 menit (Raghav, 2018; Harward et al., 2006).

4. Instrumen Pengukuran

- **Apache Benchmark (ab)** digunakan untuk melakukan 30 permintaan ke endpoint PHP dan mencatat waktu respons.
- **Logger internal** mencatat jumlah query aktual yang dilakukan ke MySQL, sebagai indikator efektivitas caching.

5. Teknik Analisis

Data waktu respons dan jumlah query dianalisis menggunakan nilai rata-rata dan standar deviasi. Validitas pengukuran diperkuat dengan replikasi eksperimen secara konsisten (Tyagi, 2023).

6. Validasi Konfigurasi

Konfigurasi Redis diuji stabilitasnya dengan simulasi beban simultan. Hasil konfigurasi Redis juga dibandingkan dengan kondisi baseline tanpa caching, sebagaimana pendekatan dalam Kusuma et al. (2017).

HASIL & PEMBAHASAN

1. Hasil Kuantitatif

Eksperimen dilakukan dengan dua skenario utama, yaitu sistem tanpa menggunakan caching Redis dan sistem

dengan Redis terintegrasi sebagai cache penyimpanan hasil query SQL. Setiap skenario diuji sebanyak 30 kali menggunakan Apache Benchmark (ab) untuk memastikan kestabilan hasil dan reliabilitas data.

Skenario	Rata-rata Waktu Respons (ms)	Jumlah Query SQL
Tanpa Redis	412	30
Dengan Redis	96	5

Tabel di atas menunjukkan bahwa penerapan Redis berhasil menurunkan waktu respons rata-rata dari 412 milidetik menjadi 96 milidetik. Penurunan ini mencapai sekitar 76%, sebuah perbedaan signifikan dalam konteks aplikasi web yang mengandalkan kecepatan respons. Tidak hanya itu, jumlah query SQL yang tereksekusi juga turun drastis dari 30 menjadi 5. Artinya, 83% permintaan berhasil dilayani langsung dari cache, tanpa harus mengakses basis data utama.

2. Pembahasan

Hasil eksperimen ini secara nyata menguatkan temuan pada berbagai studi sebelumnya yang menunjukkan efektivitas Redis sebagai mekanisme caching untuk sistem berbasis PHP dan MySQL. Penurunan waktu respons sebesar 76% tidak hanya berdampak pada pengalaman pengguna, tetapi juga mengurangi beban server dan konsumsi sumber daya. Dalam konteks arsitektur web modern, setiap milidetik sangat berharga, terutama pada aplikasi yang menangani ratusan hingga ribuan permintaan per detik.

Pengurangan jumlah query SQL yang signifikan menunjukkan Redis berhasil mengambil alih fungsi penyajian data pada permintaan berulang, yang dalam kondisi normal harus dilayani oleh MySQL. Dalam implementasi ini, Redis menyimpan hasil query dalam bentuk string JSON dengan kunci unik berdasarkan parameter permintaan. Saat permintaan dengan parameter yang sama masuk kembali, sistem langsung menyajikan data dari Redis tanpa harus mengakses database.

Redis juga memungkinkan konfigurasi **Time To Live (TTL)** sehingga data cache secara otomatis kadaluarsa setelah waktu tertentu. Dalam penelitian ini, TTL disetel selama 300 detik (5 menit) untuk menjamin bahwa data tidak terlalu lama tersimpan, sekaligus memberikan cukup waktu agar manfaat caching bisa dirasakan.

Strategi ini banyak diterapkan dalam aplikasi e-commerce, direktori, maupun sistem informasi akademik, di mana perubahan data tidak terlalu sering terjadi dalam interval pendek.

Performa Redis juga sangat stabil meskipun dijalankan dalam lingkungan virtual seperti Windows Subsystem for Linux (WSL). Ini membuktikan bahwa Redis dapat digunakan secara efektif bahkan di lingkungan pengembangan lokal, tanpa memerlukan infrastruktur server khusus. Dengan konfigurasi ringan dan integrasi mudah melalui pustaka seperti **Predis**, Redis dapat menjadi solusi caching ideal bagi pengembang PHP di berbagai skala proyek.

Namun demikian, perlu dicatat bahwa penggunaan Redis juga memiliki batasan. Konsumsi memori menjadi salah satu aspek penting yang harus diperhatikan, terutama bila data yang disimpan dalam cache berukuran besar atau terlalu banyak. Selain itu, pada sistem dengan frekuensi pembaruan data tinggi, risiko penyajian data usang dari cache dapat menjadi isu serius jika tidak dibarengi dengan strategi invalidasi yang tepat (Harward et al., 2006; Raghav, 2018).

Dari perspektif arsitektur, Redis mendukung model **cache-aside**, di mana data hanya dimasukkan ke dalam cache jika diminta. Pendekatan ini relatif aman dan fleksibel untuk skenario aplikasi yang tidak terlalu kompleks. Namun, untuk sistem skala besar, strategi seperti **write-through** atau **write-behind caching** juga perlu dipertimbangkan agar Redis tidak hanya menyimpan hasil query tetapi juga terintegrasi dalam proses pembaruan data.

KESIMPULAN

Penelitian ini menunjukkan bahwa penerapan Redis sebagai mekanisme caching in-memory pada aplikasi web PHP mampu meningkatkan performa secara signifikan. Hasil eksperimen mengungkapkan bahwa penggunaan Redis berhasil menurunkan waktu respons rata-rata aplikasi sebesar 76%, dari 412 ms menjadi 96 ms. Selain itu, penerapan caching ini juga mengurangi jumlah query ke database MySQL hingga 83%, yang menandakan bahwa Redis efektif dalam mengurangi beban query dan meningkatkan efisiensi pengambilan data.

Penerapan Redis dalam lingkungan XAMPP menunjukkan hasil yang konsisten, dengan pengurangan waktu respons yang nyata, terutama pada skenario dengan trafik tinggi dan data semi-statis. Hal ini memberikan bukti empiris bahwa Redis tidak hanya mempercepat pemrosesan data, tetapi juga mengurangi ketergantungan aplikasi pada query database yang mahal dalam hal waktu dan sumber daya.

Namun, meskipun penerapan Redis memberikan manfaat signifikan, penelitian ini juga mencatat beberapa tantangan, termasuk kebutuhan alokasi memori yang cukup pada server lokal dan risiko inkonsistensi data jika strategi invalidasi cache tidak dijalankan dengan hati-hati. Oleh karena itu, penting untuk memperhatikan pengaturan TTL dan kebijakan invalidasi data yang tepat, terutama pada aplikasi yang sering mengalami pembaruan data.

Secara keseluruhan, Redis merupakan solusi yang sangat direkomendasikan bagi pengembang aplikasi web berbasis PHP yang ingin meningkatkan performa aplikasi mereka, terutama dalam lingkungan pengembangan lokal seperti XAMPP. Studi lanjutan dapat berfokus pada penerapan Redis dalam skala yang lebih besar, seperti di lingkungan produksi dengan arsitektur mikroservis, serta evaluasi lebih mendalam terhadap teknik load balancing dan distribusi cache.

DAFTAR PUSTAKA

Alsmad, I., & Alda, S. (2012). Test cases reduction and selection optimization in testing web services. *Information Engineering and Electronic Business*, *5*, 1–8.

Arnaldo Marulitua Sinaga, & Sibarani, P. (2015). The implementation of caching database to reduce query's response time. *MATEC Web of Conferences*, 1–2.

Bora, A., & Bezboruah, T. (2015). A comparative investigation on implementation of RESTful versus SOAP-based web services. *International Journal of Database Theory and Application*, *8*(3), 297–312.

Cao, Z., Wang, Z., & Zegura, E. (2000). Performance of hashing-based schemes for internet load balancing. *Proceedings IEEE INFOCOM*, *2*, 922–931.

Chen, S., Tang, X., Wang, H., Zhao, H., & Guo, M. (2016, August). Towards scalable and reliable in-memory storage system: A case study with Redis. In *2016 IEEE Trustcom/BigDataSE/ISPA* (pp. 1660–1667). IEEE.

Creswell, J. W. (2014). *Research design: Qualitative, quantitative, and mixed methods approaches* (4th ed.). SAGE

Publications.

George, J. (2021). Optimizing database performance in web applications. *ResearchGate*. https://www.researchgate.net/publication/389710439_Optimizing_Database_Performance_in_Java_and_AngularJS_Web_Applications_Caching_and_ORM_Strategies

Gill, A., & Goyal, S. (2021). Deploying a framework to improve the performance of a web service. *International Journal of Electrical and Electronics Engineering*.

Harward, N. D., Geweke, A. R., & Voskoboynik, A. (2006). Database caching and invalidation using database-provided facilities for query dependency analysis. *United States Patent Application Publication*, 13–14.

Kusuma, M. (2016, February). Evaluasi performa web server menggunakan varnish HTTP reserve proxy dan Redis database cache. In *Seminar Nasional Inovasi dan Aplikasi Teknologi Industri* (pp. 260–264). Malang, Indonesia.

Kusuma, M., Widyawan, W., & Ferdiana, R. (2017, July). Performance comparison of caching strategy on WordPress multisite. In *3rd International Conference on Science and Technology - Computer* (pp. 176–181). Yogyakarta, Indonesia.

Li, S., Jiang, H., & Shi, M. (2017). Redis-based web server cluster session maintaining technology. In *2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)* (pp. 3065–3069). IEEE.

Macedo, T., & Oliveira, F. (2011). *Redis cookbook: Practical techniques for fast data manipulation*. O'Reilly Media Publications.

Malik, M. F., & Khan, M. N. A. (2016). An analysis of performance testing in distributed software applications. *International Journal of Modern Education and Computer Science*, *8*(7), 53–56.

Raghav, P. (2018). Improving response time for geographically distributed caches using query result [Master's thesis, National College of Ireland]. *NORMA*. https://norma.ncirl.ie/4246/1/poojaraghav.pdf

Somerville, I. (2016). *Software engineering* (10th ed.). Pearson.

Sun, T., Chen, F., & Deng, J. (2013). LVS cluster system based on cookie session maintenance. *Application Research of Computers*, *30*(4), 1102–1104.

Zulfa, M. I., Fadli, A., & Wardhana, A. W. (2020). Strategi caching aplikasi berbasis in-memory menggunakan Redis server untuk mempercepat akses data relasional. *Jurnal Teknologi dan Sistem Komputer*, *8*(2), 157–163.