BAB 1. PENDAHULUAN

1.1 Latar Belakang

Applicant Tracking System (ATS) merupakan sebuah sistem perangkat lunak yang dikembangkan untuk mendukung proses rekrutmen karyawan dengan basis data utamanya berisi informasi terkait detail pekerjaan dan calon pelamar. Pada umumnya, sistem ini beroperasi sebagai platform berbasis web yang menyediakan aksesibilitas luas bagi perekrut dan manajer personalia, sekaligus berfungsi sebagai basis data terpusat yang menampung informasi lowongan pekerjaan, profil kandidat, dokumen lamaran, serta riwayat komunikasi (Chavan *et al.*, 2024). Fungsionalitas ATS mencakup keseluruhan alur proses bisnis yang berhubungan dengan manajemen sumber daya manusia pada suatu perusahaan. ATS memungkinkan *Human Resource Manager* (HRM) untuk meningkatkan efisiensi dalam proses rekrutmen dengan mempercepat identifikasi kandidat yang sesuai kebutuhan perusahaan.

Proses perekrutan umumnya terdiri dari enam tahapan, yakni *employer branding*, *candidate attraction*, *applicant management*, *pre-selection*, *selection*, dan *hire*. Proses ini dianggap kompleks karena melibatkan pemrosesan data yang meliputi perhitungan, perbandingan, transformasi data, dan penyimpanan data baru. Dalam kondisi tertentu, pemrosesan data dapat menyebabkan waktu tunggu yang lama atau bahkan kegagalan proses akibat beban berlebih pada server. Kondisi ini menimbulkan *bottleneck* yang menghambat respons sistem terhadap permintaan pengguna pada layanan web. Hal ini bertentangan dengan tujuan utama ATS yang seharusnya responsif dalam mendukung otomatisasi, penyederhanaan tugas-tugas *Human Resource* (HR), pengurangan administrasi dan pencatatan secara manual, serta memberikan informasi dengan cepat saat diperlukan (Alhassan and Alhassan, 2025).

Dalam upaya mengatasi waktu tunggu yang panjang dan kemungkinan terjadinya *bottleneck* akibat pemrosesan data dalam jumlah besar, salah satu strategi yang tepat adalah dengan menerapkan *database caching*. *Database caching* teknik penyimpanan sementara data hasil *query database* di memori untuk mempercepat

waktu akses data pada permintaan berikutnya (Hassan *et al.*, 2022). Dengan menerapkan *caching*, beban data tidak lagi ditanggung sepenuhnya oleh server basis data utama, melainkan berbagi dengan server *database caching* yang memiliki *Input/Output Operations Per Second* (IOPS) yang tinggi, sehingga mampu meningkatkan waktu respon sistem terhadap permintaan pengguna.

Meskipun memiliki IOPS yang sangat tinggi, penerapan *database caching* menyimpan data yang bersifat sementara atau *non-persistent*, sehingga data yang disimpan di dalam *cache* akan hilang ketika terjadi gangguan infrastruktur, perbaikan, pemeliharaan dan *Force Majeure* (Mutlu *et al.*, 2023). Dalam konteks ini, ada beberapa pendekatan strategis yang dapat diterapkan untuk mengimplementasikan basis data *cache* dengan tepat. Saat ini, terdapat empat pola strategis *caching* yang umum dikenal, yaitu *Cache-Aside*, *Write-Through*, *Write-Behind*, dan *Read-Through*.

Penelitian ini akan berfokus pada implementasi, analisis perbandingan strategi *cache*, serta rekomendasi strategi yang optimal untuk ATS. Untuk memberikan rekomendasi yang tepat terkait strategi caching, penelitian ini menggunakan empat parameter evaluasi: *Cache-Hit*, *Cache-Miss*, *Response Time*, dan *Resource Utilization*. Metode *Simple Additive Weighting* (SAW) digunakan dalam pengambilan keputusan untuk menghasilkan rekomendasi yang sesuai terkait penerapan strategi *cache* dalam mengelola pemrosesan data pada ATS.

1.2 Rumusan Masalah

Mengacu pada konteks yang telah diuraikan pada bagian latar belakang, berikut merupakan perumusan masalah yang akan menjadi fokus pada penelitian ini:

1. Menghasilkan luaran berupa rekomendasi strategi yang terbaik di antara *Cache-Aside*, *Read-Through*, *Write-Through*, dan *Write-Behind* dengan metode *Simple Additive Weighting* (SAW).
2. Parameter evaluasi yang digunakan sebagai dasar rekomendasi yaitu *Cache-Hit*, *Cache-Miss*, *Response Time*, dan *Resource Utilization*.
3. Data alternatif untuk penentuan nilai preferensi diperoleh secara *real time* dari fitur *cgroup* pada sistem operasi Linux.

1.3 Tujuan

Berdasarkan rumusan masalah yang telah disebutkan, penelitian ini bertujuan untuk menerapkan teknik *cache* dan mengevaluasi serta memeringkatkan strategi *Cache-Aside*, *Write-Through*, *Write-Behind*, dan *Read-Through* dalam konteks penggunaannya pada *Applicant Tracking System*, dengan menggunakan empat kriteria evaluasi yang telah ditetapkan untuk menghasilkan rekomendasi strategi yang paling optimal.

1.4 Manfaat

Manfaat dari penelitian ini tidak hanya bersifat akademis tetapi juga praktis. Penelitian ini memberikan pemahaman melalui analisis perbandingan tentang penerapan *caching* pada *Applicant Tracking System*. Berikut merupakan manfaat dari penelitian:

1. Kontribusi pada pengetahuan, penelitian ini memberikan sebuah sudut pandang lain dalam pengembangan teknologi informasi, khususnya dalam penerapan *cache strategy* terhadap aplikasi *Human Resource Management* (HRM) yang salah satunya adalah *Applicant Tracking System* (ATS).
2. Peningkatan efisiensi *Human Resource Management* (HRM), implementasi *cache strategy* yang optimal membantu organisasi/perusahaan meningkatkan efisiensi dalam proses rekrutmen melalui optimasi beban pemrosesan data dan operasi *input/output* per detik (*IOPS*) sehingga lebih responsif.
3. Pengambilan keputusan yang lebih baik, rekomendasi yang dihasilkan dari penelitian ini membantu pengembang aplikasi untuk menentukan *cache strategy* yang sesuai dengan kebutuhan organisasi/perusahaan.