201910370311032
Akhmad Khasan Abdullah
Prodi Informatika

BAB I
PENDAHULUAN

1.1 Latar Belakang

Sistem Informasi Akademik (SIAKAD) berperan penting dalam mendukung proses administrasi dan kegiatan akademik di lingkungan sekolah, seperti pengelolaan data siswa, guru, presensi, penilaian, serta penerimaan peserta didik baru (PPDB). Penerapan SIAKAD bertujuan untuk meningkatkan efisiensi pengelolaan data dan kemudahan akses informasi bagi seluruh pemangku kepentingan [1].

Dalam pengembangan sistem informasi seperti SIAKAD, pemilihan arsitektur perangkat lunak menjadi faktor krusial karena berpengaruh langsung terhadap skalabilitas, fleksibilitas, serta kinerja sistem. Secara umum, arsitektur perangkat lunak dapat dibedakan menjadi beberapa pendekatan, di antaranya monolithic *architecture* dan *microservices architecture. Monolithic architecture* menyatukan seluruh komponen aplikasi mulai dari logika bisnis, pengelolaan data, hingga antarmuka layanan dalam satu kesatuan aplikasi [2]. Sebaliknya, microservices architecture membagi aplikasi menjadi layanan-layanan kecil yang berdiri sendiri dan saling berkomunikasi melalui mekanisme tertentu, sehingga lebih adaptif terhadap perubahan kebutuhan dan beban sistem [3].

SIAKAD di SMAN 18 Surabaya dikembangkan menggunakan arsitektur *monolithic*, di mana seluruh komponen sistem dijalankan dalam satu aplikasi terintegrasi [2]. Dalam implementasinya, sistem informasi akademik di SMAN 18 Surabaya memiliki pola beban kerja yang tidak merata. Modul tertentu seperti PPDB dan presensi mengalami lonjakan akses pada periode tertentu, misalnya awal tahun ajaran atau awal semester. Pada arsitektur *monolithic*, kondisi ini dapat menyebabkan peningkatan *response time* dan penurunan *throughput* dalam menangani permintaan secara

bersamaan karena seluruh modul saling bergantung dalam satu kesatuan aplikasi [3].

Sebagai alternatif, arsitektur *microservices* menawarkan pendekatan yang lebih fleksibel dengan memecah aplikasi menjadi layanan-layanan kecil yang bekerja secara mandiri [3]. Arsitektur ini bersifat *loosely-coupled*, mendukung skalabilitas, serta memungkinkan distribusi beban kerja yang lebih efisien [4] [5]. Penelitian sebelumnya menunjukkan bahwa arsitektur microservices memiliki keunggulan dalam menangani beban transaksi yang tinggi. Marieska M, Yunanta A, Aulia H, Utami A, Rizqie M [6] melakukan perbandingan performa antara arsitektur *monolithic* dan *microservices* pada sistem *online ticketing* dan menunjukkan bahwa *microservices* mampu memberikan waktu respon 36% lebih cepat serta menurunkan tingkat kesalahan hingga 71% pada kondisi beban tinggi. Hasil ini menunjukkan bahwa *microservices* lebih efektif dalam mengelola permintaan secara simultan dibandingkan arsitektur *monolithic*.

Pada penelitian lain yang dilakukan Rina Wati, Novita Andriyani, Priyono,Tri Susilowati [7] penelitian dilakukan dengan merancang bangun sistem informasi akademik dengan pendekatan sistem arsitektur microservices. Penelitian tersebut melakukan pengujian performa dengan membandingkan arsitektur microservices dan arsitektur monolith dengan parameter pengujian yang diukur adalah waktu respon rata-rata (ms), *throughput,* dan Tingkat kegagalan. Hasilnya menunjukkan bahwa arsitektur microservices meningkatkan performa khususnya waktu respon hingga lebih dari 50%, serta memiliki *throughput* lebih tinggi sebagai indicator bahwa sistem dapat memproses lebih banyak permintaan.

Oleh karena itu, penelitian ini bertujuan untuk mengembangkan ulang sistem SIAKAD di SMAN 18 Surabaya menggunakan arsitektur *microservices* dan membuktikan bahwa pendekatan ini mampu memberikan performa yang lebih baik dibandingkan arsitektur *monolithic*, khususnya dalam aspek *response time* dan *throughput*.

1.2 Rumusan Masalah

Berlandaskan pemaparan latar belakang yang tertera, maka rumusan masalah dalam penelitian ini adalah :

1. Bagaimana implementasi *microservices architecture* dengan *event-driven message* pada aplikasi SIAKAD di SMAN 18 Surabaya menggunakan *framework NestJS?*
2. Bagaimana pengaruh implementasi *microservices architecture* dengan *event-driven message* menggunakan *framework NestJS* terhadap performa khususnya pada parameter *response time* dan *throughput* pada aplikasi SIAKAD dibandingkan dengan *monolithic architecture* yang sebelumnya digunakan?

1.3 Tujuan Penelitian

Mengacu pada rumusan masalah yang telah dirumuskan, maka tujuan dari penelitian ini adalah :

1. Mengembangkan dan mengimplementasikan *microservices architecture* dengan *event-driven message* menggunakan *framework NestJS* pada aplikasi SIAKAD SMAN 18 Surabaya.
2. Membandingkan performa aplikasi SIAKAD berbasis microservices architecture dengan versi *monolithic architecture* dengan parameter *response time* dan *throughput*.

1.4 Batasan Masalah

Berkenaan dengan rumusan dan tujuan penelitian yang telah ditetapkan, agar penelitian ini tetap terfokus pada permasalahan yang diteliti, maka ditetapkan batasan-batasan sebagai berikut :

1. Lingkup Implementasi: Dalam penelitian ini implementasi dilakukan hanya pada aplikasi *server side* (*backend*) dalam bentuk *REST API* agar penelitian berfokus terhadap implementasi *microservices architecture* dengan *event-driven message*.
2. Framework dan Teknologi: Implementasi dilakukan menggunakan framework NestJS dengan bahasa pemograman Typescript. Penelitian tidak akan membahas framework atau bahasa lain di luar NestJS dan Typescript.
3. Aspek Performa: Analisis performa hanya mencakup perbandingan antara *monolithic architecture* yang digunakan sebelumnya dengan *microservice architecture*, terkait *average response time* dan *throughput*.
4. Event-Driven Message: Fokus terkait even-driven message dibatasi pada penggunaan message broker menggunakan NATS, dan tidak akan membahas penggunaan lain seperti RabbitMQ, Kafka, dan sebagainya.
5. Pengguna dan Fitur Aplikasi: Penelitian ini hanya mencakup fitur-fitur utama dalam modul aplikasi SIAKAD, seperti module IAM (*Identity and Access Management*), modul PPDB, dan modul fitur absensi (guru, siswa,

staff). Fitur-fitur tambahan atau pengembangan lebih lanjut di luar ini tidak akan dibahas.

6. Pengujian performa: Difokuskan pada pengujian di beberapa fungsional spesifik di beberapa modul, seperti aksi login (modul IAM), aksi daftar PPDB (modul PPDB), aksi absen (modul absesnsi). Dan dilakukan pada lingkungan *server local*.