BAB I
PENDAHULUAN

1.1 Latar Belakang

Institusi pendidikan khususnya sekolah, di era digital masa kini sedang menghadapi tantangan operasional dalam proses pengelolaan data akademik. Kehadiran Sistem Informasi Akademik (SIAKAD) menjadi penting untuk menggantikan proses manajemen data akademik yang masih terpisah dan manual.

Saat ini, mekanisme pengelolaan data akademik yang berjalan di banyak sekolah masih terpisah dan manual. Data-data akademik seperti data siswa, guru, dan nilai sering kali tersimpan di berbagai media penyimpanan yang tidak saling terhubung seperti Google Drive dan berkas fisik. Kondisi ini memaksa pihak sekolah untuk menuliskan data dalam bentuk berkas (kertas), dan juga melakukan *input* data pada aplikasi pengolah angka (*spreadsheet*). Hal ini meningkatkan risiko kesalahan manusia (*human error*) serta data yang menjadi tidak konsisten.

Kehadiran *Software as a Service* (*SaaS*) dengan arsitektur *multi-tenant* dapat menjadi solusi untuk sentralisasi data akademik antar sekolah, di mana satu layanan aplikasi mampu melayani banyak *tenant*. Efektivitas arsitektur ini sangat bergantung pada strategi penyimpanan data, antara model *Database-per-Tenant* yang memisahkan basis data antar *tenant* dengan isolasi data yang absolut namun memboroskan sumber daya, atau model *Shared Database* yang menawarkan efisiensi sumber daya namun menuntut mekanisme isolasi data yang kompleks. Tantangan utamanya adalah bagaimana merancang arsitektur *multi-tenant* dengan model *Shared Database* yang efisien, namun tetap mampu menjamin keamanan isolasi data yang setara dengan model *Database-per-Tenant*. Dengan itu, arsitektur ini memunculkan dua masalah yang dihadapi, yaitu jaminan isolasi data dan stabilitas performa.

Masalah pertama adalah kegagalan isolasi data. Berdasarkan hasil wawancara yang penulis lakukan di lima sekolah, ditemukan bahwa 4 dari 5 sekolah masih menggunakan platform Google Drive dan berkas fisik untuk menyimpan data akademik yang bersifat sensitif. Praktik ini menyoroti ketiadaan jaminan keamanan dan isolasi data yang sistematis. Pada arsitektur *multi-tenant* Kumar et al, (2025) mengonfirmasi bahwa "tantangan signifikan terkait keamanan data, privasi, dan isolasi" adalah risiko inheren. Kegagalan isolasi ini dapat berujung pada "*cross-tenant access violations*", sebuah risiko yang terbukti nyata melalui insiden seperti kerentanan "*ChaosDB*" di Azure yang secara tidak sengaja memberikan akses basis data antar *tenant*.

Masalah kedua adalah kegagalan performa. Berdasarkan hasil wawancara yang penulis lakukan, 2 dari 5 sekolah yang telah mengadopsi sistem digital seringkali mengalami kelambatan (*lag*) parah yang terjadi disaat jam sibuk, seperti saat *input* data nilai secara massal. Literatur akademis secara jelas mengidentifikasi ini sebagai "*Noisy Neighbour*" atau "*performance interference*". Secara spesifik, Lange et al, (2020) membuktikan secara kuantitatif bahwa *bottleneck* utama adalah persaingan pada penyimpanan (I/O). Penelitian mereka menunjukkan bahwa beban kerja tulis (*write*) yang berat dan konkuren dapat menyebabkan "degradasi performa 3 kali hingga 5 kali" bagi semua pengguna lain yang berbagi sumber daya.

Hingga saat ini, penelitian untuk mengatasi "*Noisy Neighbour*" cenderung berfokus pada solusi tingkat infrastruktur yang kompleks. Sebagai contoh, Xiao et al, (2023) mengusulkan sistem "YISHAN" di Alibaba Cloud, yang menggunakan *Machine Learning* untuk memprediksi pola beban dan secara dinamis memigrasi seluruh *database instance* antar *host* fisik untuk menemukan "*good packing*". Namun, solusi tingkat infrastruktur seperti ini tidak praktis dan tidak dapat diakses oleh pengembang aplikasi *SaaS* pada umumnya yang tidak memiliki kendali atas *host* atau router jaringan.

Oleh karena itu, terdapat kesenjangan penelitian untuk menemukan solusi mitigasi "*Noisy Neighbor*" yang praktis dan dapat diimplementasikan langsung di tingkat arsitektur aplikasi (*SaaS*). Penelitian ini mengusulkan arsitektur komprehensif yang menjawab kedua masalah tersebut, yaitu menjamin isolasi data yang ketat menggunakan *Global Query Filters (GQF)*, dan memitigasi *Noisy Neighbor* menggunakan pola *Queue-Based Load Leveling,* Pola antrean ini divalidasi secara teoretis oleh Sun et al, (2025) sebagai arsitektur yang ideal untuk memisahkan "produksi" (generasi) pesan yang cepat dari "konsumsi" (pemrosesan) yang lambat, sehingga mengisolasi beban kerja tulis yang berat dan menjaga stabilitas sistem bagi semua *tenant*.

1.2 Rumusan Masalah

Berdasarkan latar belakang dan identifikasi masalah yang telah dipaparkan, rumusan masalah yang akan dikaji dalam penelitian ini adalah:

1. Bagaimana merancang arsitektur *Software as a Service* (*SaaS*) *multi-tenant* berbasis *Shared Database* yang efisien, namun tetap menjamin keamanan isolasi data yang setara dengan model *Database-per-Tenant*?
2. Bagaimana mengimplementasikan mekanisme *Global Query Filters* (*GQF*) pada model *Shared Database* untuk menjamin isolasi data dan mencegah akses lintas *tenant* (*cross-tenant access*)?
3. Bagaimana mengimplementasikan pola *Queue-Based Load Leveling* untuk memitigasi masalah *Noisy Neighbor*?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah tersebut, penelitian ini dilaksanakan dengan tujuan untuk:

1. Membangun arsitektur *SaaS* berbasis *Shared Database* yang mampu menjamin keamanan isolasi data yang setara dengan model *Database-per-Tenant*.
2. Mengimplementasikan *Global Query Filters* (*GQF*) pada arsitektur sistem, sehingga mencegah akses lintas entitas (*cross-tenant access)*.
3. Mengimplementasikan dan mengevaluasi kinerja pola arsitektur *Queue-Based Load Leveling* dalam memitigasi masalah *Noisy Neighbor*.

1.4 Manfaat Penelitian

Hasil penelitian ini diharapkan membawa manfaat sebagai berikut:

1. Manfaat Praktis

a. Bagi Pengembang Perangkat Lunak

Menyediakan model arsitektur yang konkret dan teruji menggunakan teknologi umum (ASP.NET Core, Microsoft SQL Server) sebagai panduan praktis untuk membangun aplikasi *SaaS multi-tenant* yang lebih stabil, andal, dan *scalable*, khususnya dalam mengatasi tantangan isolasi data dan *Noisy Neighbor*.

b. Bagi Institusi Pendidikan (Sekolah)

Hasil dari penerapan arsitektur ini berpotensi memberikan pengalaman penggunaan SIAKAD yang lebih lancar dan konsisten, mengurangi gangguan akibat kelambatan sistem saat jam sibuk, sehingga mendukung efektivitas proses akademik.

2. Manfaat Akademis

a. Menambah wawasan ilmu pengetahuan mengenai teknik isolasi performa dalam konteks arsitektur *multi-tenant*, khususnya pada model *Shared Database* yang sering digunakan namun rentan terhadap interferensi.

b. Memberikan bukti empiris mengenai efektivitas penerapan pola *Queue-Based Load Leveling* di tingkat aplikasi (ASP.NET Core) sebagai solusi mitigasi *Noisy Neighbor*.

1.5 Batasan Masalah

Penelitian ini membatasi ruang lingkup pembahasan pada area tertentu guna menjaga fokus studi, yaitu:

1. Penelitian ini berfokus pada perancangan dan pengujian arsitektur yang secara spesifik menangani dua tantangan pada lingkungan *multi-tenant*:
   a) Isolasi Data.
   b) Stabilitas Performa (*Noisy Neighbor*).
2. Model basis data yang digunakan adalah *Shared Database* dengan pendekatan *Shared Schema*.
3. Arsitektur dibangun menggunakan *stack* teknologi ASP.NET Core untuk *backend* dan Microsoft SQL Server untuk manajemen basis data.
4. Arsitektur hanya mencakup fungsi inti SIAKAD (misal: operasi nilai/pendaftaran) yang penting untuk simulasi *read* dan *write*. Aspek *UI/UX* dan fungsionalitas lengkap (seperti manajemen keuangan) dikesampingkan.
5. Efektivitas arsitektur dievaluasi melalui simulasi *load testing* (latensi, *throughput*) dan pengujian fungsional, bukan di lingkungan produksi nyata.