BAB V
KESIMPULAN DAN SARAN

5. 1 Kesimpulan

Berdasarkan hasil dan pembahasan, penelitian ini berhasil melakukan dekomposisi dan perancangan sistem Tracer Study dengan menerapkan arsitektur *microservices.* Perancangan dilakukan dalam 3 tahap, yaitu mempelajari sistem *monolithic* lama, mendekomposisi sistem lama menggunakan *Domain-Driven Design (DDD),* dan mendefinisikan spesifikasi setiap *service.* Hasil perancangan tersebut dieksekusi dan diimplementasikan hingga sistem dapat digunakan. Eksekusi dan implementasi dilakukan dalam 3 tahap, yaitu eksekusi pengembangan *service,* mengintegrasikan *microservices* dengan sistem lama dan membangun *API Gateway,* serta melakukan pengujian fungsional setiap *service* dengan melakukan *integration testing* dan *performance testing.*

Hasil analisis dari *performance testing* pada sistem *monolithic* dan *microservices* menunjukkan pengaruh penerapan *microservices* dari sisi kinerja sistem. Ditemukan bahwa penerapan sistem *microservices* meningkatkan performa sistem dari sisi *response time* dari rata-rata 1138ms menjadi 154ms. Selain itu, *throughput* atau jumlah *request* yang dapat diproses per hari juga meningkat, dari rata-rata 1 *request* per detik (8400 *request* per hari) menjadi 16,4 *requests* per detik (1.419.960 *requests* per hari). Persentase kegagalan atau *error rate* pada penggunaan normal sistem *microservices* juga lebih rendah dibanding sistem *monolithic,* di mana *error rate* pada sistem *monolithic* rata-rata sebesar 0,23% sedangkan pada sistem *microservices* rata-rata sebesar 0,00%.

5. 2 Saran

Dari penelitian ini terdapat beberapa saran dan masukan yang dapat dijadikan referensi pada penelitian-penelitian selanjutnya, di antaranya:

1. Melakukan validasi ahli, *tim developer* sistem *monolithic,* dan validasi pengguna terhadap sistem yang dikembangkan. Metode validasi dapat berupa angket kuesioner atau wawancara untuk menilai kesesuaian fungsionalitas

hasil migrasi dengan sistem sebelumnya. Dari validasi ini juga terdapat kemungkinan ditemukannya *bug* yang terlewat saat pengujian *internal,* sehingga perbaikan dapat dilakukan dan sistem dapat lebih *mature.*

2. Melakukan diskusi dengan *domain expert* saat menentukan kandidat *microservices* dengan menggunakan *Domain-Driven Design.* Hasil diskusi ini dapat membantu peneliti untuk dapat membuat rancangan *microservices* yang lebih baik.
3. Melakukan pengujian apakah *microservices* yang dikembangkan telah memenuhi ekspektasi dari *driving forces* yang telah ditetapkan sebelum pengembangan. Pengujian ini dapat menentukan dilakukannya pengembangan *microservices* lain di masa mendatang.
4. Melakukan perhitungan dan pembandingan *cost* atau biaya pengembangan dan pemeliharaan saat menerapkan arsitektur *monolithic* dengan saat menerapkan arsitektur *microservices.*
5. Mengevaluasi hasil penerapan sistem *microservices* dalam jangka waktu tertentu dan pengaruhnya terhadap produktivitas tim pengembang.