BAB V
KESIMPULAN

5.1 Kesimpulan

Berdasarkan penelitian analisis performan REST API Golang Gin dan Node.js Express pada skema database SQL/NoSQL pada platform shortener yang telah dilakukan, maka dapat disimpulkan bahwa:

a. Dua API dikembangkan menggunakan *framework* Golang Gin dan Node.js Express, masing-masing terhubung ke Postgresql dan MongoDB. Kedua API memiliki 16 endpoint identik dan di-deploy di Google Cloud Platform dengan konfigurasi serupa.
b. Pengujian melibatkan 9 skenario dengan tiga tahapan utama—breakpoint, load, dan spike testing—menggunakan K6. Pengujian ini memberikan gambaran kinerja setiap kombinasi *framework* dan *database* untuk API URL Shortener.
c. Secara keseluruhan, Golang Gin menunjukkan performa lebih baik daripada Node.js Express, dengan nilai median lebih tinggi dalam *breakpoint testing* (15 vs. 14) dan efisiensi *cpu* serta memori yang lebih baik selama *load* dan *spike testing*, menandakan pemilihan *framework* berpengaruh terhadap performa API yang dikembangkan.
d. Postgresql menunjukkan performa lebih unggul dibandingkan MongoDB, baik dalam *breakpoint testing* (median 26,5 vs. 8) maupun efisiensi CPU dan memori selama *load* dan *spike testing*, menandakan

pemilihan *database* berpengaruh terhadap performa API yang dikembangkan.

e. Golang dengan Postgresql menghasilkan performa terbaik, dengan median tertinggi dalam breakpoint testing (32). Dalam pengujian semua skenario sistem pada iterasi per detik tertinggi yang didapatkan, Golang Postgresql hasilkan tingkat keberhasilan terbaik dengan 99% keberhasilan dalam *load* dan *spike testing* serta memberikan performa yang stabil.
f. Dalam pengujian *end to end* dengan login, kombinasi Golang Postgresql menunjukkan performa paling stabil dengan waktu pemrosesan terbaik pada load dan spike testing, baik dalam metrik iterasi per detik maupun permintaan per detik.

5.2 Saran

Terdapat beberapa saran yang harus diperhatikan dalam penelitian ini yaitu seperti:

a. Menambahkan jenis pengujian performa seperti *soak testing* yang dapat melihat bagaimana performa sistem jika dijalankan pada beban tertentu secara konstan pada waktu yang lama.
b. Membuat versi *frontend* untuk mengakomodasi API yang sudah dikembangkan sehingga sistem url shortener dapat digunakan secara penuh.