Pengujian Performa API (Application Programming Interface) dengan Metode Load Testing

Gede Herdian Setiawan[1], I Made Budi Adnyana[2], Komang Budiarta[3]
Sistem Komputer
Institut Teknologi dan Bisnis ITB STIKOM Bali
Denpasar, Indonesia
e-mail: [1]herdian@stikom-bali.ac.id, [2]budi.adnyana@stikom-bali.ac.id, [3]komang_budiarta@stikom-bali.ac.id

Abstrak

Aplikasi smart Activity Tracking merupakan aplikasi yang dikembangkan pada platform mobile Android berbasis LBS (Location Base Services) yang mampu terintegrasi dengan berbagai platform dengan menerapkan API (Application Programming Interface). Untuk memastikan performa API, penelitian ini melaukan pengujian Load testing pada setiap endpoint API. pengujian load testing berfokus pada tiga indikator yaitu response time, throughtput dan error rate. pengujian dilakukan dengan menggunakan tools JMeter pada 5 endpoint API dengan variasi beban pada masing-masing endpoint. Hasil menunjukkan berdasarkan response time dan throughtput performa pada API terdapat penurunan seiring dengan bertambahnya beban request dari user namun setiap endpoint masing mampu menangani request dengan baik ditunjukkan dengan error rate 0.0%. Dengan demikian performa aplikasi Smart Activity Tracking sangat baik ditunjukkan dengan tidak adanya request yang gagal diproses pada setiap endpoint.

Kata kunci: *API (Application Programming Interface), Software Testing, Performa Testing, Load Testing, Apache JMeter.*

Abstract

Smart Activity Tracking application is an application developed on the Android mobile platform based on LBS (Location Base Services) which is integrated with various platforms by implementing an API (Application Programming Interface). To ensure API performance, this research performs Load testing on each API endpoint. testing load testing focuses on three indicators namely response time, throughput and error rate. testing is done using JMeter tools on 5 API endpoints with load variations on each endpoint. The results based on the response time and throughput performance on the API there is a decrease as the demand load from the user increases but each endpoint is able to handle requests well with an error rate of 0.0%. Thus, the Smart Activity Tracking application is very well demonstrated with no disappointing requests at any endpoint..

Keywords*: API (Application Programming Interface), Software Testing, Performance Testing, Load Testing, Apache JMeter.*

1. Pendahuluan

Aplikasi smart Activity Tracking merupakan aplikasi yang dikembangkan pada *platform mobile* Android berbasis LBS *(Location Base Services)* untuk keperluan pemantauan aktivitas Mahasiswa di lingkungan kampus sebagai upaya penerapan physical distancing untuk mengendalikan penularan Covid-19. Aplikasi menerapakan Teknologi Geofencing untuk memberikan area pemantauan secara *real time* pada area tertentu sehingga pemantauan aktivitas bisa dilakukan secara langsung. Aplikasi mampu terintegrasi dengan berbagai *platform* dengan didukung oleh teknologi API *(Application Programming Interface)*.

API merupakan sebuah teknologi yang dikembangkan dengan tujuan agar sebagian atau seluruh fungsi sistem dapat digunakan secara terprogram [1]. API bisa dikatakan sebuah interface yang dapat menghubungkan aplikasi pada lintas platform. API terdiri dari gabungan perintah yang membentuk library dan berfungsi untuk interaksi antar aplikasi. API disebut juga sebagai Web Service yang menggunakan protokol HTTP. [2] API pada sebuah aplikasi dituntut memiliki performa yang handal karena harus mampu

melayani berbagai request dari berbagai aplikasi antar platform. Untuk memastikan performa API perlu dilakukan berbagai tahapan testing seperti Load testing [3].

Load Testing merupakan teknik performance testing dimana respon sistem diukur dalam berbagai kondisi dan beban. Pengujian ini membantu menentukan bagaimana software berperilaku ketika beberapa user mengakses software secara bersamaan pengujian menggunakan tipe Load Testing digunakan untuk memeriksa bagaimana sebuah sistem yang sedang dikembangkan bisa menangani masalah atau beban yang diujikan yang disesuaikan dengan keadaan sebenarnya [4],[5].

Pengujian load testing telah dilakukakan oleh Arlinta dkk, hasil pengujian Load Test API pada perangkat lunak yang dikembangkan diperoleh bahwa performa oleh sebagian besar API mengalami penurunan seiring dengan bertambahnya jumlah user yang mengakses [6]. Difa dkk, melakukan Load testing dengan melihat *response time*, *error rate* dan juga *throughput* serta menganalisis arsitektur untuk menentukan performa Microserfice yang lebih baik [7].

Makalah ini bermaksud melakukan pengujian performa API pada aplikasi smart activity tracking dan memberikan analisa performa API sebagai refrensi pengembangan aplikasi. Pengujian performa dengan *load testing* berfokus pada tiga indikator yaitu (i) *response time* : waktu rata-rata respon pada API, (ii) *throughput* : penanganan *request* pada sever, (iii) *Error rate* : presentase error pada saat penanganan request berlangsung.

2. Metode Penelitian

Gambar 1. Metode Penelitian

Pada Gambar 1 menunjukkan metode usulan yang terdiri dari beberapa tahapan, antara lain : persiapkan, tahap pengujian dan tahap Analisa hasil.

2.1. Persiapan

Persiapkan dilakukan untuk menentukan *test plant* dan mempersiapkan *endpoint* pada API yang akan diuji. *Test plant* dan *endpoint* API yang diuji ditunjukan pada Tabel 1.

Tabel 1. Test plant (load testing)

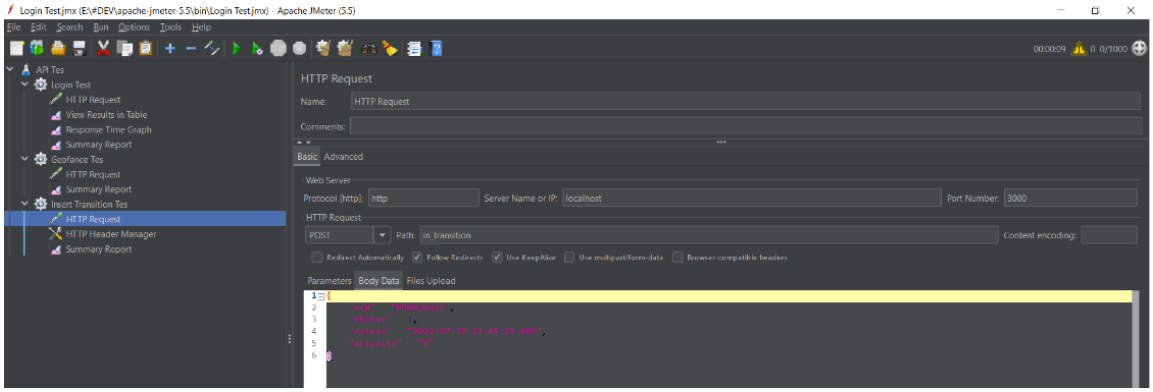
No	*API*		*Tes Plant Load Testing*
	Parameter	***Endpoint***	***Number of thread***
1	POST	*login*	10,100,500,1000
2	GET	*geofance*	10,100,500,1000
3	POST	*insert_transition*	10,100,500,1000
4	POST	*update_transition*	10,100,500,1000

Load testing dilakukan pada ke lima *endpoint* API dengan beban disesuaikan pada setiap *endpoint*.

2.2. Pengujian

Pengujian *load testing* berfokus pada tiga indikator yaitu : Response time merupakan ukuaran waktu tunggu saat *client (user)* melakukan request ke server hingga *server* memberikan *response* kembali, *Throughput* merupakan jumlah *request user* yang diproses *server* dalam satuan detik / *request per second (rps)*, dan *Error rate* merupakan perhitungan persentase proses yang gagal saat proses *request* dan *response* berlangsung [8].

Tahap pengujian dilakukan dengan aplikasi JMeter. JMeter digunakan untuk melakukan simulasi beban pada *server*, untuk menguji performa atau menganalisis performa secara menyeluruh. JMeter merupakan aplikasi berbasis *desktop* dengan plaform Java VM yang digunakan untuk mengukur kinerja perangkat lunak *client-server* melalui *load* tes. JMeter dapat menguji perangkat lunak pada berbagai bahasa pemrograman seperti PHP, JSP, ASP dan bahasa pemrograman lainnya [9],[10]. Gambar 2 menujukkan tampilan konfigurasi JMeter pada penelitian ini.



Gambar 2. Konfigurasi JMeter

2.3. Analisa Hasil

Analisa hasil dilakukan pada tahap akhir pengujian, analisa hasil bertujuan untuk mengetahui performa API secara menyeluruh berdasarkan beban dan *response time* pada masing – masing *endpoint*.

3. Hasil dan Pembahasan

3.1. Hasil pengujian Load Testing

Pengujuan load testing dilakukan dengan menggunakan tools pada aplikasi JMeter, fokus pengamatan pada tiga indikator yaitu : *Response time*, *Throughput* dan *Error Rate* pada masing-masing *endpoint* API dengan kombinasi beban *request* untuk dapat mengetahui performa setiap *endpoint*. Hasil pengujian load testing ditunjukan pada Tabel 2.

Tabel 2. Hasil Pengujian *Load Testing*

Endpoint	***Number of thread (user)***	***Response time (ms)***	***Throughput (rps)***	***Error Rate***
login	10	32	1.1	0.0%
login	100	53	10.0	0.0%
login	500	56	49.5	0.0%
login	1000	70	97.2	0.0%
geofance	10	45	1.1	0.0%
geofance	100	59	9.9	0.0%
geofance	500	61	48.7	0.0%
geofance	1000	108	93.9	0.0%
insert_transition	10	48	1.1	0.0%
insert_transition	100	54	53.7	0.0%
insert_transition	500	51	49.8	0.0%
insert_transition	1000	55	23.6	0.0%
update_transition	10	51	1.5	0.0%
update_transition	100	64	59.0	0.0%
update_transition	500	69	65.2	0.0%
update_transition	1000	73	68.8	0.0%

3.2. Analisa Hasil Pengujian

Berdasarkan hasil pengujian *load testing* seperti yang ditujukan pada Tabel 2, performa setiap *endpoint* API pada ketiga indikator sangat berkaitan dengan beban *request* pada *user (client)*. Pada indikator *response time* hasilnya menunjukan rata-rata *response time* dalam satuan waktu *(ms)* berbading lurus dengan jumlah beban *request*, semakin besar beban maka rata-rata *response time* meningkat. Pada indikator *throughput* menujukan semakin besar beban *request* maka waktu proses pada *server* dalam satuan detik meningkat hal ini mengakibatkan beban semakin meningkat menyebabkan performa API menurun namun semua *request* masih bisa ditangani yang ditunjukan pada error rate 0.0% pada setiap proses.

4. Kesimpulan

Penelitian ini dilakukan untuk mengukur performa pada Aplikasi Smart Activity Tracking melalui *Load testing* pada setiap *endpoint* API. pengujian load testing berfokus pada tiga indikator yaitu *response time, throughtput* dan *error rate*. pengujian dilakukan dengan menggunakan *tools* JMeter pada lima *endpoint* API dengan variasi beban pada masing-masing *endpoint*. Hasil menunjukkan berdasarkan *response time* dan *throughtput* performa pada API terdapat penurunan seiring dengan bertambahnya beban *request* dari user namun setiap *endpoint* masing mampu menangani *request* dengan baik ditunjukkan dengan *error rate* 0.0%. Dengan demikian performa aplikasi Smart Activity Tracking sangat baik

ditunjukkan dengan tidak adanya *request* yang gagal*(error)* diproses pada setiap *endpoint*. Untuk penelitian berikutnya perlu dilakukan uji fungsionalitas pada antarmuka aplikasi sehingga performa aplikasi dapat diketahui secara menyeluruh..

Daftar Pustaka

[1] J. Wang, X. Bai, H. Ma, L. Li, and Z. Ji, "Cloud API Testing," in *2017 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2017, pp. 385–386, doi: 10.1109/ICSTW.2017.71.

[2] S. M. Sohan, C. Anslow, and F. Maurer, "SpyREST: Automated RESTful API Documentation Using an HTTP Proxy Server (N)," in *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2015, pp. 271–276, doi: 10.1109/ASE.2015.52.

[3] Cahaya Purtri Agustika, W. Syaifullah JS, and M. Idhom, "Pengujian Aplikasi Greenwallet Dengan Metode Load Testing Dan Apache Jmeter," *J. Inform. dan Sist. Inf.*, vol. 2, no. 2, pp. 190–195, 2021, doi: 10.33005/jifosi.v2i2.357.

[4] D. I. Permatasari, "Pengujian Aplikasi menggunakan metode Load Testing dengan Apache JMeter pada Sistem Informasi Pertanian," *J. Sist. dan Teknol. Inf.*, vol. 8, no. 1, p. 135, 2020, doi: 10.26418/justin.v8i1.34452.

[5] D. Intan Permatasari and B. Santoso, "Pengukuran Throughput Load Testing Menggunakan Test Case Sampling Gorilla Testing," *Semin. Nas. Sist. Inf.*, no. 4, pp. 2008–2014, 2019.

[6] Arlinta Christy Barus, Johannes Harungguan, and Efren Manulu, "Pengujian Api Website Untuk Perbaikan Performansi Aplikasi Ditenun," *J. Appl. Technol. Informatics Indones.*, vol. 1, no. 2, pp. 14–21, 2022, doi: 10.54074/jati.v1i2.33.

[7] D. Al Fansha, M. Y. H. Setyawan, and M. N. Fauzan, "Load Test pada Microservice yang menerapkan CQRS dan Event Sourcing," *J. Buana Inform.*, vol. 12, no. 2, p. 126, 2021, doi: 10.24002/jbi.v12i2.4749.

[8] S. A. Kamarudin, Kusrini, "Uji kinerja sistem web service pembayaran mahasiswa menggunakan apache jmeter (studi kasus: Universitas amikom yogyakarta)," *Teknol. Inf.*, vol. XIII, no. 1, pp. 44–52, 2018.

[9] J. Wang and J. Wu, "Research on Performance Automation Testing Technology Based on JMeter," in *2019 International Conference on Robots & Intelligent System (ICRIS)*, 2019, pp. 55–58, doi: 10.1109/ICRIS.2019.00023.

[10] S. Radhakrishna and M. Nachamai, "Performance inquisition of web services using soap UI and JMeter," in *2017 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC)*, 2017, pp. 1–5, doi: 10.1109/ICCTAC.2017.8249993.