COMPARISON OF NOCODB AND NOCOBASE PERFORMANCE EFFECTIVENESS IN DEVELOPING ELECTRONIC-BASED GOVERNMENT SYSTEM APPLICATIONS

PERBANDINGAN EFEKTIVITAS PERFORMA NOCODB DAN NOCOBASE DALAM PENGEMBANGAN APLIKASI SISTEM PEMERINTAHAN BERBASIS ELEKTRONIK

Shelly Kurnia Ulisyah[1], *Briansyah Setio Wiyono[2]
[1,2] Universitas Muhammadiyah Malang, Jl. Raya Tlogomas No.246, Malang
shellykurul@webmail.umm.ac.id[1], brian@umm.ac.id[2]
[*] *corresponding author*

Abstract – *The implementation of government digitalization through the Electronic-Based Government System (SPBE) requires a fast and reliable application development platform. The low-code platform is an efficient solution as it allows for application development with minimal coding. This research compares two open-source low-code platforms, NocoDB and NocoBase, focusing on performance effectiveness in supporting SPBE application development. Testing was conducted using K6 load testing with a configuration of 100 virtual users over the same duration. The results show that NocoDB has a higher throughput of 92.1 requests per second with a total of 2,763 requests, though accompanied by 86 failed checks, indicating response fluctuations under high load. In contrast, NocoBase recorded 33.3 requests per second with 100 successful requests and no failures, demonstrating more consistent response stability despite lower throughput. Thus, NocoDB is more effective for high-load scenarios, while NocoBase excels in service stability. These results are expected to serve as a technical reference in selecting the optimal low-code platform for the implementation of digital government systems based on SPBE.*

Keywords - *system effectiveness; K6; low-code; NocoBase; NocoDB*

Abstrak – Penerapan digitalisasi pemerintahan melalui Sistem Pemerintahan Berbasis Elektronik (SPBE) membutuhkan platform pengembangan aplikasi yang cepat dan andal. Platform low-code menjadi solusi efisien karena memungkinkan pembangunan aplikasi dengan sedikit penulisan kode. Penelitian ini membandingkan dua platform low-code open-source, yaitu NocoDB dan NocoBase, dengan fokus pada efektivitas performa dalam mendukung pengembangan aplikasi SPBE. Pengujian dilakukan menggunakan K6 load testing dengan konfigurasi 100 virtual users dalam durasi yang sama. Hasil menunjukkan bahwa NocoDB memiliki throughput lebih tinggi, yaitu 92,1 permintaan/detik dengan 2.763 total permintaan, namun disertai 86 failed checks, menandakan fluktuasi respons di bawah beban tinggi. Sebaliknya, NocoBase mencatat 33,3 permintaan/detik dengan 100 permintaan berhasil tanpa gagal, menunjukkan stabilitas respons yang lebih konsisten meskipun throughput lebih rendah. Dengan demikian, NocoDB lebih efektif untuk skenario beban tinggi, sedangkan NocoBase lebih unggul dalam kestabilan layanan. Hasil ini diharapkan menjadi referensi teknis dalam pemilihan platform low-code yang optimal untuk implementasi sistem pemerintahan digital berbasis SPBE.

Kata Kunci - efektivitas sistem; K6; low-code; NocoBase; NocoDB

I. PENDAHULUAN

Penerapan digitalisasi pemerintahan melalui Sistem Pemerintahan Berbasis Elektronik (SPBE) merupakan strategi penting dalam mendorong efisiensi, transparansi, dan akuntabilitas layanan publik. SPBE menekankan integrasi antar sistem dan proses bisnis berbasis teknologi informasi yang saling terhubung serta berkelanjutan. Salah satu tantangan utama dalam implementasinya adalah bagaimana mengembangkan sistem yang cepat, efisien, dan adaptif terhadap perubahan kebutuhan tanpa mengorbankan performa teknis [1]. Platform low-code dan no-code menjadi salah satu solusi strategis untuk menjawab tantangan tersebut. Keduanya memungkinkan pengembangan aplikasi dengan sedikit atau tanpa penulisan kode, sehingga dapat mempercepat proses pembangunan sistem pemerintahan digital [2]. Dua platform open-source yang digunakan saat ini adalah NocoDB dan NocoBase, yang keduanya menawarkan pendekatan visual untuk manajemen basis data dan pembuatan antarmuka [3].

NocoDB menekankan otomatisasi dari basis data relasional seperti MySQL dan PostgreSQL, sedangkan NocoBase mengadopsi pendekatan modular berbasis plugin untuk fleksibilitas pengembangan [4]. Perbedaan arsitektur ini menjadikan keduanya menarik untuk dibandingkan, terutama dalam konteks pengembangan aplikasi SPBE yang menuntut kinerja tinggi serta keandalan waktu respons. Sejumlah penelitian terdahulu telah menyoroti manfaat platform *low-code* dari sisi produktivitas dan kemudahan penggunaan, namun masih terbatas yang membahas efektivitas teknis berdasarkan pengujian performa sistematis dengan simulasi pengguna nyata [5][6]. Padahal, dalam sistem pemerintahan, keandalan teknis seperti waktu respons, stabilitas, dan *throughput* sangat krusial karena melibatkan ribuan pengguna secara bersamaan [7]. Untuk mengukur efektivitas teknis tersebut, digunakan alat K6, yaitu *framework* open-source untuk load testing yang mampu mensimulasikan ribuan pengguna paralel serta mengukur metrik seperti waktu respons, *error rate, dan throughput* [8]. K6 telah terbukti efektif dalam pengujian REST API di berbagai studi evaluasi performa sistem, termasuk aplikasi publik [9].

Namun, penelitian terdahulu belum menjelaskan secara eksplisit hubungan antara karakteristik modul SPBE (misalnya autentikasi pengguna, pengelolaan arsip, atau unggah berkas) dengan profil beban dan efektivitas platform *low-code* yang digunakan. Selain itu, belum terdapat perumusan hipotesis operasional yang menguji apakah NocoDB lebih unggul pada skenario CRUD massal**,** sementara NocoBase lebih stabil pada skenario autentikasi atau form submission**.** Melalui pendekatan kuantitatif-eksperimental**,** penelitian ini bertujuan untuk memberikan gambaran empiris mengenai performa kedua platform dalam kondisi uji yang terukur dan terstandar[10]. Dengan begitu, hasilnya dapat digunakan sebagai dasar dalam menentukan platform yang lebih sesuai untuk kebutuhan implementasi SPBE.

Selain relevan dengan kebutuhan efisiensi sistem digital, pemilihan kedua platform juga mempertimbangkan sifat open-source dan lisensi terbukanya yang sesuai untuk pengembangan layanan pemerintahan [11]. Penelitian ini juga didukung oleh dokumentasi teknis dan komunitas aktif dari masing-masing platform yang menyediakan panduan pengujian berbasis Docker [12]. Pendekatan evaluasi ini mengacu pada prinsip pengukuran kuantitatif performa perangkat lunak yang menekankan konsistensi hasil uji [13], serta mengikuti studi terkait rekayasa kualitas perangkat lunak modern [14].Dengan demikian, penelitian ini diharapkan dapat memperluas referensi ilmiah mengenai pemanfaatan teknologi *low-code open-source* di sektor publik**,** sekaligus memberikan dasar empiris untuk perencanaan dan evaluasi kapasitas sistem pemerintahan digital[15].

II. SIGNIFIKANSI STUDI

A. Studi Literatur

Penelitian ini berlandaskan studi terdahulu tentang pengembangan sistem informasi low-code, evaluasi efektivitas teknis, dan pengujian performa sistem, sebagai dasar kerangka evaluasi NocoDB dan NocoBase dalam SPBE. *Low-code* memungkinkan pengembangan cepat dan partisipasi pengguna non-programmer [1], namun tetap memerlukan pengujian performa sistematis [2]. Pendekatan eksperimen komparatif digunakan dengan data primer untuk menilai *throughput, response time, dan error rate*. Mengacu ISO/IEC 25010:2011, efisiensi performa mencakup *time behavior, resource utilization, dan capacity* [4]. K6 dipilih untuk simulasi pengguna virtual dan pengukuran metrik performa, sementara load testing menilai kemampuan sistem menghadapi beban tinggi [8]. Pemilihan NocoDB dan NocoBase didasarkan dokumentasi teknis dan lisensi terbuka yang sesuai pengembangan SPBE [11]. Penelitian ini mengisi celah kajian empiris dan menyediakan data kuantitatif sebagai pertimbangan instansi pemerintah dalam memilih platform low-code.

B. Tujuan dan Desain Penelitian

Penelitian ini membandingkan efektivitas performa dua platform *low-code open-source*, NocoDB dan NocoBase, dalam pengembangan SPBE. Evaluasi dilakukan melalui pengujian beban menggunakan K6 dengan 100 Virtual Users selama 30 detik di lingkungan Docker, untuk memperoleh data primer performa sistem secara paralel. Desain eksperimen komparatif ini memastikan kondisi pengujian seragam, meski jenis endpoint berbeda. Hasil uji dikumpulkan dalam bentuk numerik dan visual untuk menilai kecepatan, stabilitas, dan efisiensi respons platform. Pendekatan ini menekankan pentingnya *load testing* dan *capacity planning*, sekaligus memberikan gambaran kuantitatif efektivitas performa sistem *low-code* untuk aplikasi publik [15].

C. Prosedur Penelitian

Penelitian dilakukan secara sistematis untuk memastikan konsistensi pengujian performa kedua platform (Gambar 3). Proses dimulai dengan instalasi NocoDB dan NocoBase menggunakan Docker, diikuti konfigurasi sistem dan pembuatan basis data uji yang serupa agar perbandingan adil. Skrip pengujian performa dibuat dengan K6 (*JavaScript*) dan dijalankan dengan 100 *Virtual Users* selama 30 detik, menggunakan ambang batas p(95)<500 ms dan rate<0,1. Setiap *request* API dievaluasi berdasarkan status respons dan waktu pemrosesan, dengan cuplikan skrip untuk masing-masing platform disediakan berikut.

```
import http from 'k6/http';
import { check, sleep } from 'k6';
import { htmlReport } from "https://raw.githubusercontent.com/benc-uk/k6-reporter/main/dist/bundle.js";

export const options = {
    vus: 100,
    duration: '30s',
    thresholds: {
        'http_req_duration': ['p(95)<500'],
        'http_req_failed': ['rate<0.1'],
    },
};
export default function () {
    const url = 'http://localhost:8080/api/v1/db/data/v1/produk'; // Endpoint GET NocoDB

    const res = http.get(url);
    check(res, {
        'status is 200': (r) => r.status === 200,
        'response time < 500ms': (r) => r.timings.duration < 500,
    });
    sleep(1);
}
export function handleSummary(data) {
    return {
        "summary-get-nocodb.html": htmlReport(data),
    };
}
```

Gambar 1. Cuplikan skrip K6 untuk penguji NocoDB

Dan ini untuk skrip pengujian pada Nocobasenya seperti ini.

```
import http from 'k6/http';
import { check, sleep } from 'k6';
import { htmlReport } from "https://raw.githubusercontent.com/benc-uk/k6-reporter/main/dist/bundle.js";

export const options = {
    vus: 100,
    duration: '30s',
    thresholds: {
        'http_req_duration': ['p(95)<500'],
        'http_req_failed': ['rate<0.1'],
    },
};
export default function () {
    const url = 'http://localhost:8080/api/v1/db/data/v1/EGA%20SPBE/Login';

    const params = {
        headers: {
            'xc-token': 'efJdKmJjSeW7Uki8iRH9cImnjfEfZZfTVXPDUVi8UY',
        },
    };
    const res = http.get(url, params);
    check(res, {
        'status is 200': (r) => r.status === 200,
        'response time < 500ms': (r) => r.timings.duration < 500,
    });

    sleep(1);
}
export function handleSummary(data) {
    return {
        "summary-nocodb.html": htmlReport(data),
}
```

Gambar 2. Cuplikan skrip K6 untuk penguji NocoBase

Selanjutnya, hasil pengujian dari masing-masing platform berupa summary report HTML yang berisi data numerik (jumlah request, waktu respons, dan tingkat kegagalan). Laporan ini kemudian diekstraksi dan dibandingkan untuk memperoleh perbedaan efektivitas performa antar platform. Setiap hasil pengujian diinterpretasikan berdasarkan indikator metrik performa yang telah ditetapkan sebelumnya, meliputi *response time*, *throughput*, *error rate*, dan *failed checks*. Data hasil uji dianalisis secara kuantitatif untuk menentukan sejauh mana efektivitas sistem dalam menangani beban virtual user. Tahapan keseluruhan proses penelitian digambarkan pada Gambar 3 berikut:

Gambar 3. Diagram alur metode penelitian

D. Struktur Variabel dan Indikator Efektivitas

Penelitian ini menggunakan dua jenis variabel, yaitu:

- Variabel Bebas
  Platform yang diuji, yaitu NocoDB dan NocoBase.
- Variabel Terikat
  Efektivitas performa teknis yang diukur melalui empat indikator utama:
  1. Response Time (ms) – waktu rata-rata sistem merespons permintaan; semakin kecil nilainya semakin baik.
  2. Request Rate (request/second) – jumlah permintaan yang diproses per detik; semakin tinggi nilainya semakin baik.
  3. Error Rate (jumlah kesalahan) – persentase permintaan yang gagal; semakin rendah nilainya semakin stabil.

4. Efisiensi Data (MB/request) – rata-rata data yang digunakan per permintaan; semakin kecil nilainya semakin efisien.

Keempat indikator ini mewakili dimensi *performance efficiency* sesuai standar ISO/IEC 25010:2011.

E. Spesifikasi Perangkat

Untuk spesifikasi perangkat yang digunakan dalam pengujian ini adalah:

- Perangkat keras
  - CPU: Intel Core i5 / AMD Ryzen 5
  - RAM: Minimal 8 GB
  - Penyimpanan: SSD 256 GB
  - Sistem Operasi: Windows 10
- Perangkat Lunak
  - Docker & Docker Compose
  - K6 Load Testing
  - Chrome/Firefox

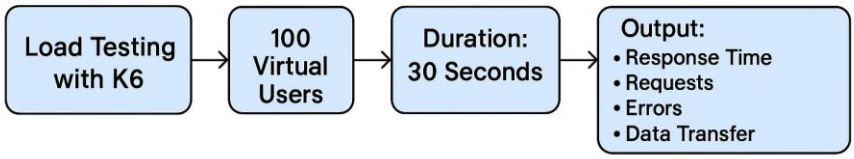
F. Metode Pengujian

Pengujian pada kedua platform tersebut menggunakan alat K6, yaitu load testing berbentuk script pemrograman bahasa JavaScript untuk mengukur performa layanan berbasis HTTP. Alat ini dapat memungkinkan pengguna untuk mengatur jumlah yang akan di simulasikan dan sesuai dengan jumlah yang sudah dijelaskan sebelumnya, pengujian dilakukan dengan mensumulasikan 100 virtual user untuk mengakses endpoint yang ditentukan selama 30 detik tanpa adanya jeda *(constant load)*. Setiap virtual users (VUs) akan mengirimkan permintaan secara berulang untuk mensimulasikan user nyata yang intensif.

Parameter teknis yang digunakan dalam pengujian meliputi:

- Durasi Pengujian: 30 detik untuk setiap skenario endpoint
- Jumlah Virtual Users (VUs): 100 pengguna simultan
- Thresholds (Ambang Batas Kinerja):
  - *Response Time*: 95% dari semua permintaan harus selesai dalam waktu kurang dari 500 milidetik
  - *Error Rate*: tingkat kegagalan harus di bawah 10% dari total permintaan

Output dari pengujian ini dihasilkan dalam bentuk laporan numerik dan grafik ringkasan. Pengujian dijalankan dalam lingkungan Docker untuk memastikan stabilitas konfigurasi sistem, serta mencegah gangguan eksternal seperti fluktuasi jaringan atau ketergantungan eksternal lainnya. Berikut skema pengujian secara umum.



Gambar 4. Skema Pengujian menggunakan K6

G. Teknik Analisis Data

Data hasil pengujian performa menggunakan alat K6 dianalisis dengan pendekatan kuantitatif deskriptif. Tujuannya untuk memberikan gambaran objektif mengenai kemampuan masing-masing platform dalam menangani beban permintaan secara simultan. Langkah-langkah analisis dilakukan melalui beberapa tahapan sebagai berikut:

1. Pengelompokan Data

Data uji dibedakan berdasarkan:

- Jenis platform: NocoDB dan NocoBase.

- Jenis protokol permintaan: GET dan POST, yang mewakili interaksi umum sistem dalam konteks pengujian performa.

2. Perhitungan Metrik Efektivitas Performa Metrik utama yang digunakan dalam analisis meliputi:

- Request Rate

  Untuk rumusnya perhitungannya = $\frac{Total\ Request}{Durasi\ pengujian}$ (1)

  Mengukur kemampuan sistem dalam memproses permintaan per detik.
- Error Rate (Failed Checks)

  Untuk rumus perhitungannya = $\left(\frac{Failed\ checks}{Total\ checks}\right) x\ 100\%$ (2)

  Menilai stabilitas respon sistem terhadap permintaan yang dikirimkan.
- Tingkat Kepatuhan Waktu Respons

  Untuk rumus perhitungannya = $\left(\frac{Cek\ Passed}{Total\ Checks}\right) x\ 100\%$ (3)

  Mengukur sejauh mana permintaan berhasil dijawab sesuai batas waktu
- Efisiensi Data

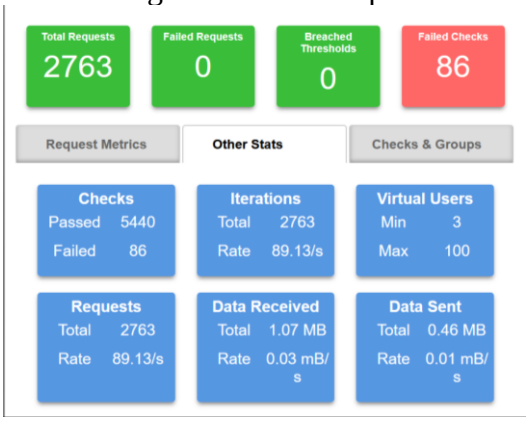
  Untuk rumus perhitungannya = $\frac{Total\ Data\ yang\ dikirim + diterima\ (MB}{Total\ Requests}$ (4)

Mengukur efisiensi penggunaan bandwidth per permintaan.
Setiap metrik yang diperoleh digunakan untuk menilai efektivitas sistem dari segi kecepatan, kestabilan waktu respons, efisiensi penggunaan data, dan ketahanan terhadap beban akses paralel. Interpretasi dilakukan secara menyeluruh untuk menentukan platform mana yang lebih unggul dalam konteks pengembangan aplikasi SPBE.

III. HASIL DAN PEMBAHASAN

A. Hasil Uji Performa NocoDB

Pengujian terhadap NocoDB dilakukan menggunakan dua skenario metode permintaan yang dijalankan melalui alat uji *K6*, dengan simulasi 100 Virtual Users (VUs) selama 30 detik. Tujuan utama pengujian ini adalah menilai efektivitas sistem dalam merespons beban akses pengguna secara paralel. Skrip pengujian dijalankan dalam bahasa pemrograman JavaScript untuk memastikan konsistensi skenario uji.
Gambar 5 menampilkan hasil *summary report* dari uji performa NocoDB. Berdasarkan hasil tersebut, tercatat bahwa total permintaan yang berhasil diproses mencapai 2763 request, tanpa adanya *failed request* (0). Namun, terdapat 86 *failed checks* dari total 5526, yang menunjukkan bahwa sebagian kecil permintaan tidak memenuhi ambang batas waktu respons < 500 ms.

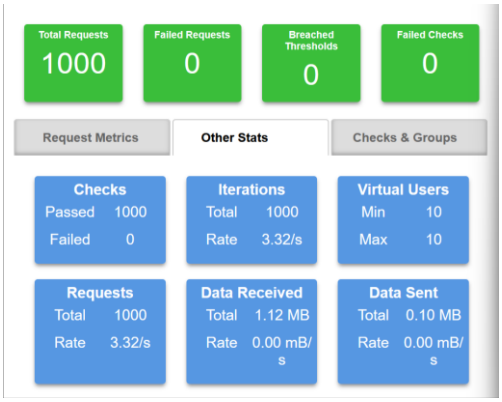


Gambar 5. Hasil Uji K6 pada NocoDB

Hasil pengujian menunjukkan NocoDB berhasil menangani 2763 request tanpa gagal, namun tercatat 86 failed checks dari 5526 checks, karena response time >500 ms. Parameter lainnya:request rate 89,13 req/s, data received 1,07 MB (0,03 MB/s), data sent 0,46 MB (0,01 MB/s), dengan virtual users 3–100. Hal ini menandakan meski semua request diproses, waktu respons NocoDB kurang stabil di beban tinggi. Sebaliknya, NocoBase tidak memiliki failed requests maupun failed checks, menunjukkan kestabilan waktu respons pada setiap percobaan login.

B. Hasil Pengujian NocoBase

Pengujian performa NocoBase difokuskan pada endpoint login (POST /api/v1/signin) untuk menilai efektivitas sistem dalam menangani autentikasi pengguna pada SPBE. Pengujian menggunakan K6 dengan 10 virtual users (VUs) selama 30 detik, mensimulasikan autentikasi paralel moderat. Skrip K6 JavaScript mengirimkan permintaan login berulang dengan payload email dan password, serta memeriksa respons menggunakan fungsi check() untuk memastikan status 200 OK dan waktu respons <500 ms. Hasil eksekusi ditampilkan dalam laporan visual (Gambar 5), memberikan insight performa sistem terhadap beban uji.



Gambar 6. Hasil Uji K6 pada NocoBase

Parameter teknis yang diamati menunjukkan hasil sebagai berikut:

- Request Rate: 89.13 request/detik — menggambarkan kemampuan sistem menangani beban tinggi dalam waktu singkat.
- Data Received: 1.07 MB dengan laju 0.03 MB/s.
- Data Sent: 0.46 MB dengan laju 0.01 MB/s.
- Virtual Users: 3–100 selama pengujian berlangsung.

Meskipun tidak terdapat permintaan yang gagal secara keseluruhan, hasil *failed checks* menunjukkan adanya variasi waktu respons di atas ambang batas performa. Hal ini menandakan bahwa pada kondisi beban tinggi, sistem NocoDB masih mampu memproses seluruh permintaan, tetapi menunjukkan tingkat kestabilan waktu respons yang fluktuatif. Nilai ini menjadi salah satu acuan untuk analisis efektivitas performa pada subbab berikutnya..

C. Pembahasan Hasil Komparatif

Perbandingan performa antara NocoDB dan NocoBase disajikan dalam tabel berikut sebagai rangkuman dari hasil pengujian menggunakan K6 load testing:

TABEL 1
PERBANDINGAN HASIL UJI PERFORMA NOCODB DAN NOCOBASE

Aspek Pengujian	NocoDB	NocoBase
Jumlah Virtual Users	100	100
Durasi Pengujian	30 detik	30 detik
Total Requests	2.763	1.000
Failed Requests	0	0
Failed Checks	86	0
Checks (Lulus/Gagal)	5.440 / 86	1.000 / 0
Iterations	2.763	1.000

Hasil pengujian menunjukkan NocoDB memiliki throughput lebih tinggi tetapi tercatat 86 failed checks, menandakan sebagian respons melebihi ambang <500 ms, kemungkinan akibat efisiensi query, serialisasi data, atau cold cache. NocoBase lebih stabil tanpa kegagalan meski jumlah permintaan lebih sedikit, cocok untuk sistem yang mengutamakan konsistensi waktu tanggap. Dari sisi efisiensi data, NocoDB memproses lebih banyak per siklus, sedangkan NocoBase lebih hemat bandwidth. Secara ringkas, NocoDB unggul dalam kapasitas dan kecepatan, NocoBase lebih stabil dan efisien sumber daya.

D. Analisis Metrik Efektivitas

Efektivitas sistem dalam konteks pengujian ini diukur menggunakan tiga indikator utama: Request Rate (Throughput), Waktu Respons Terpenuhi (Success Rate Berdasarkan Threshold), Efisiensi Data. Keempsat indikator ini dirumuskan untuk memberikan gambaran kuantitatif terhadap performa sistem dalam skenario pengujian aktual menggunakan alat *load testing* K6.

- Untuk indikator pertama yaitu Request Rate (Throughput) menggunakan perhitungan pada rumus (1)

$$RR = \frac{TR}{D}$$

Keterangan:
RR: Request Rate (request/detik)
TR: Total Requests
D: Durasi Pengujian (detik)
Hasil:
NocoDB: RR = 2763 / 30 = 92.1 request/detik
NocoBase: RR = 1000 / 30 = 33.3 request/detik
Hasil ini menunjukkan bahwa NocoDB mampu memproses hampir tiga kali lebih banyak permintaan dibandingkan NocoBase dalam waktu yang sama, menandakan keunggulan dari sisi *throughput*.

- Untuk indikator kedua yaitu Request Rate (Throughput) menggunakan perhitungan pada rumus (2)

$$TKT = \left(\frac{CP}{TC}\right) x\ 100\%$$

Keterangan:
TKT: Tingkat Kepatuhan Threshold
CP: Cek yang Passed
TC: Total Cek (Passed + Failed)

Hasil:
NocoDB: TKT = 5440 / 5526 × 100% ≈ 98.4%
NocoBase: TKT = 2000 / 2000 × 100% = 100%

Artinya, Perhitungan ini menandakan keunggulan throughput NocoDB. Metrik tool pengujian sedikit lebih rendah dibagian 89,13 request/detik karena hanya menghitung request berhasil dan menggunakan averaging, sehingga perhitungan manual bersifat teoritis sedangkan metrik tool mencerminkan performa nyata sistem.

Untuk indikator ketiga yaitu Tingkat Kepatuhan Threshold menggunakan perhitungan pada rumus (3)

$$ER = \left(\frac{FC}{TC}\right) x\ 100\%$$

Keterangan:
ER = Error Rate
FC = Failed Checks
TC = Total Checks
Hasil:
NocoDB: ER = 86 / 5526 × 100% ≈ 1.56%

NocoBase: ER = 0 / 2000 × 100% = 0%
Meskipun keduanya tidak mencatat *failed requests* secara langsung, namun NocoDB mengalami *failed checks* akibat keterlambatan respon, sementara NocoBase menunjukkan hasil sempurna dalam semua uji performa.

- Untuk indikator keempat yaitu Efisiensi Data menggunakan perhitungan pada rumus (4)

$$ED = \frac{TD}{TR}$$

Keterangan:
ED: Efisiensi Data (MB/request)
TD: Total Data (diterima + dikirim)
TR: Total Requests

Hasil:
NocoDB: TD = 1.07 MB + 0.46 MB = 1.53 MB → ED ≈ 0.00055 MB/request
NocoBase: TD = 1.12 MB + 0.10 MB = 1.22 MB → ED = 0.00122 MB/request
NocoDB lebih efisien dalam penggunaan data per request dibandingkan NocoBase, yang berarti dalam satu permintaan, sistem menggunakan *bandwidth* lebih kecil, meskipun beban datanya lebih tinggi secara total.

E. Pembahasan Hasil Metrik Efektivitas

Berdasarkan hasil pengujian, terlihat bahwa NocoDB dan NocoBase memiliki keunggulan pada aspek yang berbeda. NocoDB menunjukkan performa throughput yang lebih tinggi, dengan jumlah permintaan yang jauh lebih banyak dalam waktu yang sama. Namun, performa tinggi ini diimbangi dengan tingginya jumlah *failed checks*, yang menandakan ketidakstabilan waktu respons dalam kondisi beban tinggi. Sebaliknya, NocoBase menunjukkan kestabilan yang sangat baik, tanpa adanya permintaan atau pemeriksaan yang gagal. Meskipun jumlah permintaannya lebih sedikit dan throughput-nya rendah, seluruh respons yang dikembalikan stabil di bawah ambang batas waktu yang ditetapkan. Berikut adalah ringkasan perbandingan metrik performa antara kedua platform:

TABEL 2
PERBANDINGAN METRIK PERFORMA NOCODB DAN NOCOBASE

Metrik	NocoDB	NocoBase
Total Requests	2.763	100
Virtual Users	100	30 detik
Request Rate (Throughput)	92.1 req/detik	33.3 req/detik
Failed Requests	86	0
Kepatuhan Waktu Respons	98.4%	100%
Error Rate	1.56%	0%
Total Data (Diterima + Dikirim)	1.53 MB	1.22 MB
Efisiensi Data (MB/request)	0.00055	0.000122

IV. KESIMPULAN

Penelitian ini membandingkan efektivitas dua platform low-code, NocoDB dan NocoBase, dalam pengembangan sistem pemerintahan berbasis elektronik (SPBE) melalui uji beban menggunakan K6. Berdasarkan hasil pengujian dan analisis metrik performa, diperoleh bahwa NocoDB unggul dalam kapasitas pemrosesan data dengan throughput yang lebih tinggi dan penggunaan data per permintaan yang lebih efisien. Sementara itu, NocoBase menunjukkan kestabilan waktu respons yang lebih konsisten, tanpa adanya gagal uji (*failed checks*) selama pengujian berlangsung. Perbedaan hasil ini menunjukkan bahwa kedua platform memiliki kekuatan pada aspek yang berbeda. NocoDB lebih sesuai untuk skenario dengan beban akses data tinggi, sedangkan NocoBase lebih cocok untuk skenario yang membutuhkan konsistensi performa dan efisiensi sumber daya jaringan**.** Untuk memperkuat hasil penelitian, disarankan adanya pengujian lanjutan dengan variasi beban dan skema pengguna yang lebih terkontrol**,** serta perluasan analisis terhadap aspek keamanan, skalabilitas jangka panjang, dan integrasi sistem. Dengan demikian, penelitian ini dapat menjadi acuan awal bagi pengembang dan instansi pemerintahan dalam menentukan platform low-code yang paling efektif untuk kebutuhan aplikasi SPBE.

REFERENSI

[1] H. Nam, T. Nam, M. Oh, and S. Choi, "An Efficiency Measurement of E-Government Performance for Network Readiness: Non-Parametric Frontier Approach," *J. Open Innov. Technol. Mark. Complex.*, vol. 8, pp. 1–19, 2022.

[2] S. F. A. Razak, Y. P. Ernn, F. I. Yussoff, U. A. Bukar, and S. Yogarayan, "Enhancing Business Efficiency through Low-Code/No-Code Technology Adoption: Insights from an Extended UTAUT Model," *J. Human, Earth, Futur.*, vol. 5, no. 1, pp. 85–99, 2024.

[3] M. B. A. Andra, E. H. Hermaliani, A. S. Subekti, and M. H. Haris, "Pengenalan Big Data Dan Pembuatan Aplikasi Database Dengan NocoDB Pada Badan Santunan Yatim Pondok Cina Depok," *CONSEN Indones. J. Community Serv. Engagem.*, vol. 4, no. 2, pp. 183–189, 2024.

[4] Y. Watanabe, Y. Anang, and M. Takahashi, "Quality Model and Quality Characteristics Evaluation Suitable for Software 2.0," *Int. J. Eng. Technol. Innov.*, vol. 14, no. 3, pp. 309–320, 2024.

[5] M. A. Oktafianto, B. T. Hanggara, and M. A. Akbar, "Analisis Perbandingan Performa Framework Web Server Nest JS dan Hapi JS Berbasis REST API," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 9, no. 2, pp. 1–10, 2025.

[6] NocoBase Team, "NocoBase vs NocoDB: An In-Depth Comparison of Open-Source No-Code Tools Publication/Website: Medium," Medium.

[7] O. I. Kadhm and A. H. Hamad, "High Transaction Rates Performance Evaluation for Secure E-government Based on Private Blockchain Scheme," *Al-Khwarizmi Eng. J.*, vol. 19, no. 3, pp. 44–55, 2023.

[8] M. D. Fuady and N. Sutisna, "Analisis Kinerja Mikroservis dan Monolitik Menggunakan K6 pada Sistem Booking Tutor," *J. TEKNO KOMPAK*, vol. 20, no. 1, pp. 65–75, 2025.

[9] A. GODINHO, J. ROSADO, F. SA, and F. CARDOSO, "Performance Comparison of RESTful Web APIs using a Test Suite: .NET vs. Java Spring Boot," *J. Softw. Syst. Dev.*, vol. 2024, pp. 1–32, 2024.

[10] I. Online, "Sarcouncil Journal of Engineering and Computer Sciences An Empirical Study on Low-Code Platforms for Business Process Automation in Hybrid Cloud Environments," vol. 0, pp. 655–661, 2025.

[11] F. H, "NocoDB," GitHub.

[12] D. Docs, "Install Docker Compose," Docker Documentation.

[13] B. Wang, D. Li, and S. Zhang, "The performance quantitative model based on the specification and relation of the component," *Mathematics*, vol. 7, no. 8, pp. 1–14, 2019, doi: 10.3390/math7080730.

[14] S. R. C. C. T. Tadi, "Scalability and Performance Benchmarking of Low-Code Platforms vs. Traditional Development in Large-Scale Enterprise Applications," *J. Sci. Eng. Res.*, vol. 8, no. 8, pp. 262–273, 2021.

[15] M. Neelapu, "The Effectiveness of Load and Performance Testing on Application Scalability," *ESP J. Eng. Technol. Adv.*, vol. 4, no. 3, pp. 171–180, 2024.