Perancangan Back-End Server Menggunakan Arsitektur Rest dan Platform Node.JS (Studi Kasus: Sistem Pendaftaran Ujian Masuk Politeknik Negeri Ujung Pandang)

Ahsan Mubariz[1)], Dahlia Nur[2)], Eddy Tungadi[3)], Muhammad Nur Yasir Utomo[4)]
[1,2,3,4] Teknik Elektro, Politeknik Negeri Ujung Pandang
ahsan.mubariz@gmail.com
dahlia@poliupg.ac.id
eddy.tungadi@poliupg.ac.id
yasirutomo@gmail.com

Abstrak

Back-end merupakan program yang berjalan pada sisi *server* untuk berinteraksi langsung dengan basis data dan melaksanakan proses *logic* dari suatu sistem *web*. Salah satu teknologi antarmuka aplikasi atau layanan dari suatu program adalah *Application Programming Interface* (API). API untuk layanan web yang menggunakan protokol HTTP adalah *Representational State Transfer* (REST). Penggunaan REST API sebagai *back-end* layanan web memungkinkan layanan web diakses oleh sistem lain tanpa batasan bahasa, *environment,* maupun *platform* dari sisi *front-end*. Sistem *existing* Ujian Masuk Politeknik Negeri Ujung Pandang (UMPN) memiliki kendala pada waktu respons saat banyaknya akses secara bersamaan. Untuk itu, pada penelitian ini *back-end* dari sistem UMPN dibangun ulang menggunakan arsitektur REST dan platform Node.js. Node.js memiliki keunggulan pada teknik *non-blocking* yang memungkinkan operasi-operasi dijalankan secara paralel, sehingga memungkinkan banyak *request* dapat diselesaikan secara paralel. Fitur-fitur dari sistem yang dibangun telah diuji dan berjalan dengan kinerja yang baik. Adapun hasil uji kinerja dari sistem yang telah dibangun menunjukkan peningkatan waktu respons pada kondisi pengujian dengan akses dari *virtual users* yang meningkat dengan rata-rata peningkatan sebesar 52% pada skenario yang diujikan. Adapun sistem yang dibuat memiliki kinerja yang lebih baik dibandingkan sistem *existing* berdasarkan hasil uji dengan rata-rata selisih waktu respons berdasarkan skenario pengujian sebesar 3222,5 ms.

Keywords: *Back-end, Node.js, REST API, UMPN.*

I. PENDAHULUAN

Back-end merupakan suatu program yang berjalan pada sisi server (server-side) yang melakukan tugas untuk berinteraksi langsung dengan basis data dalam melakukan manipulasi data ke basis data, sehingga back-end tidak melakukan interaksi secara langsung kepada pengguna [1]. Dalam interaksi antar layanan client yang berbeda, salah satu teknologi yang digunakan adalah Application Programming Interfaces (API). API merupakan antarmuka yang digunakan untuk mengakses aplikasi atau layanan dari sebuah program[2]. API memungkinkan developer untuk memakai fungsi yang sudah ada dari aplikasi lain sehingga tidak perlu membuat ulang dari awal.

Salah satu arsitektur back-end adalah Representational State Transfer (REST). REST merupakan seperangkat prinsip arsitektur yang melakukan transmisi data melalui antarmuka yang terstandardisasi seperti HTTP. REST API sendiri merupakan istilah yang dipakai untuk layanan web yang mengimplementasikan arsitektur REST sebagai API[3].

Untuk membangun sebuah back-end server, diperlukan penggunaan bahasa pemrograman yang berjalan pada sisi server (server-side). Penelitian sebelumnya yang terkait dengan pembangunan back-end server seperti yang dilakukan oleh I. B.P. Manuaba dan E. Rudiastini yang membahas sistem back-end dengan API REST pada Sistem Informasi Asesmen Dosen di Politeknik Negeri Bali [4]. Sistem tersebut menjalankan layanan REST API menggunakan bahasa pemrograman hypertext preprocessor (PHP). Selain itu, penelitian komparatif dilakukan oleh Anugerah Christian Rompis dan Rizal Fathoni Aji, mengatakan layanan REST API yang mengutamakan respons cepat tanpa perhitungan yang banyak REST API berjalan di atas platform Node.js lebih sesuai [5] .

Node.js merupakan platform yang dibangun di atas runtime JavaScript Chrome untuk membangun aplikasi dengan cepat serta mudah untuk diskalakan [6]. Node.js sendiri memiliki keunggulan pada teknik *non-blocking* yang memungkinkan operasi-operasi dijalankan oleh sistem secara paralel tanpa harus menunggu operasi sebelumnya selesai sehingga memungkinkan banyak request dapat diselesaikan secara parallel [7].

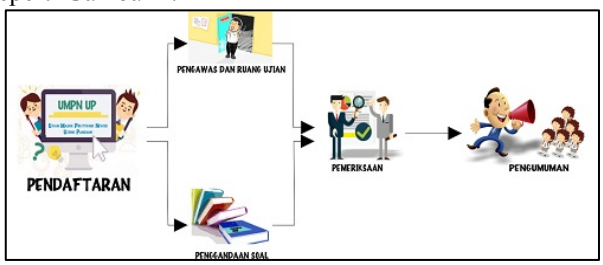
Politeknik Negeri Ujung Pandang (PNUP) telah menerapkan sistem ujian masuk secara online. Kepala Unit Pelaksana Teknis (UPT) Server, Syahril Syam, S.Kom., M.T. mengatakan bahwa sistem existing memiliki kendala pada waktu respons. Pernyataan ini didasari oleh banyaknya akses saat batas akhir waktu pendaftaran. Saat dilakukan pengujian dari dua sisi, server dan aplikasi, menunjukkan adanya permasalahan pada sisi aplikasi. Teknologi yang digunakan tidak lagi mampu untuk menampung data yang semakin banyak dalam waktu bersamaan [8].

Untuk itu, pada penelitian ini diusulkan perancangan sistem back-end dengan arsitektur REST API untuk dengan studi kasus sistemz pendaftaran Ujian Masuk Politeknik Negeri Ujung Pandang menggunakan platform Node.js dan membandingkan kinerjanya dengan sistem sebelumnya.

II. KAJIAN LITERATUR

A. Sistem Ujian Masuk Politeknik Negeri Ujung Pandang

Sistem pendaftaran online UMPN di PNUP secara umum terdiri dari blok pendaftaran, pengawas dan ruang ujian, penggandaan soal, pemeriksaan, dan pengumuman seperti Gambar 1.



Gambar 1 Sistem Pendaftaran UMPN Secara Umum[9]

Penelitian terkait dengan sistem pendaftaran UMPN Politeknik Negeri Ujung Pandang dilakukan oleh M. Nur Yasir Utomo . Penelitian tersebut mengembangkan Sistem UMPN Politeknik Negeri Ujung Pandang yang sebelumnya berupa *website* statik ke Aplikasi *Web Content Management System* (CMS). Sistem yang dikembangkan menggunakan bahasa pemrograman *Hypertext Preprocessor* (PHP) versi 5 dan *database* MySQL [10]. Penelitian Selanjutnya dilakukan oleh Ayu Angreini, dkk dengan menambahkan modul pendaftaran secara *online* pada sistem yang telah ada [9].

Sistem sebelumnya memiliki kendala pada waktu respons saat banyak akses saat batas akhir waktu pendaftaran berdasarkan wawancara dengan kepala unit pelaksana teknis (UPT) Server PNUP, Syahril Syam, S.Kom., M.T.. Saat dilakukan pengujian dari dua sisi, server dan aplikasi, menunjukkan adanya permasalahan pada sisi aplikasi. Teknologi yang digunakan tidak lagi mampu untuk menampung data yang semakin banyak dalam waktu bersamaan [8].

B. Back-End Server

Secara konsep, *back end* merupakan pembagian unit dari suatu aplikasi, terutama aplikasi berbasis *web* dan *mobile.* Unit yang termasuk pada sisi *back end* adalah keseluruhan bagian yang berjalan pada sisi server (*server-side*) [1].

C. REST API

REST (*REpresentational State Transfer*) merupakan standar arsitektur layanan komunikasi berbasis web yang. REST pada umumnya menggunakan protokol komunikasi data HTTP (*Hypertext Transfer Protocol*). REST diperkenalkan ke publik pertama kali oleh Roy Fielding pada tahun 2000 [11].

D. MongoDB

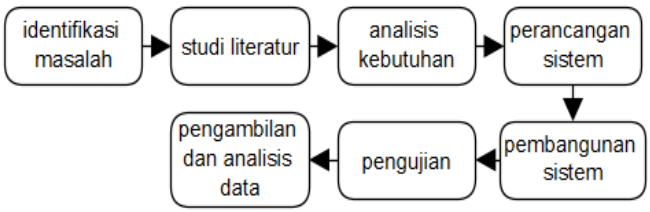
MongoDB (dari kata "*humongous*") merupakan sebuah *document oriented database* yang bersifat *open source*. MongoDB menggunakan konsep NoSQL, yang merupakan suatu konsep basis data non-*relational*. Istilah NoSQL merupakan akronim dari "N*ot Only SQL*" yang merupakan *database* yang memiliki sistem manajemen yang berbeda dari *database* relasional dalam beberapa cara [12]. MongoDB bersifat *schema-less,* yang artinya pada MongoDB tidak mengenal baris, kolom, maupun tabel.

E. Node.js

Node.js merupakan sebuah perangkat lunak yang di desain untuk pengembangan perangkat lunak. Node.js sendiri dieksekusi pada sisi server. Platform ini menggunakan bahasa pemrograman JavaScript. Selain itu, Node.js juga menggunakan teknik *nonblocking* untuk mempercepat proses.

III.METODE PENELITIAN

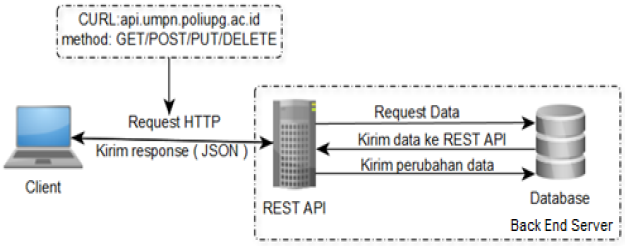
Metode dalam penelitian diperlukan agar penelitian lebih terstruktur, sehingga hasil yang diperoleh sesuai dengan tujuan pada penelitian. Adapun tahapan metode penelitian seperti pada Gambar 2.



Gambar 2. Metode Penelitian

A. Perancangan Sistem

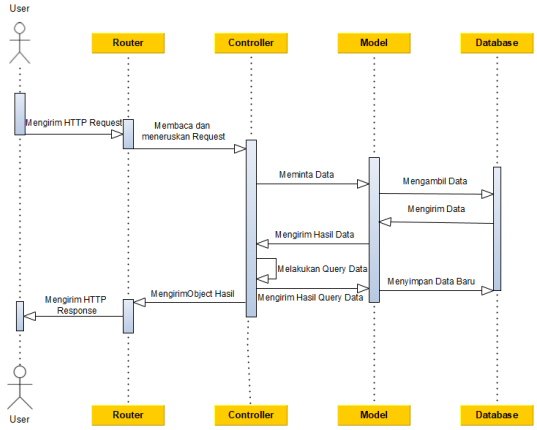
1. Gambaran Umum Sistem



Gambar 3. Gambaran Umum Sistem

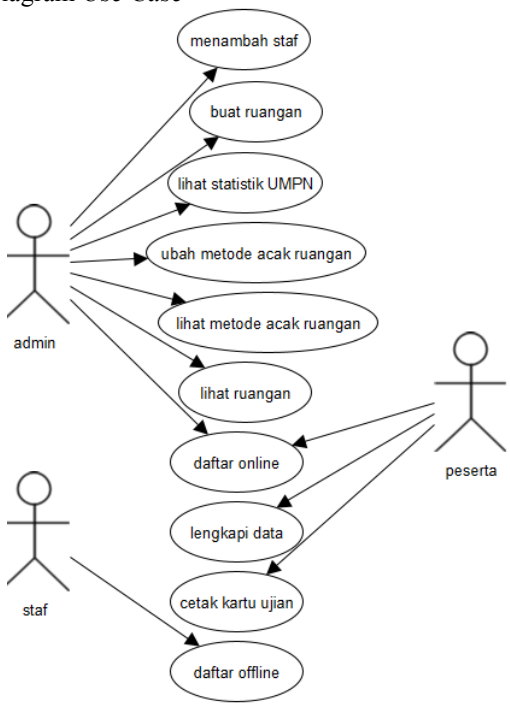
Berdasarkan gambar 3, dapat dilihat bahwa untuk memperoleh data dari *database*, client harus mengirimkan request data ke back-*end server* dengan metode CURL. *Request* kemudian diterima oleh REST API yang kemudian melakukan tindak lanjut sesuai dengan jenis *request* dari *client*. REST API kemudian mengambil data *existing* dari *database*. Data yang telah diterima kemudian diproses oleh subprogram yang telah dibuat sesuai dengan request. Jika permintaan client hanya untuk melihat data yang ada, maka REST API mengirimkan *response* ke *client* tanpa mengubah data pada *database*. Jika layanan yang diminta oleh *client* kemudian menghasilkan perubahan terhadap data sebelumnya pada *database*, hasil *query* data pada REST API dikembalikan ke *database* untuk memperbarui data yang sebelumnya. Kemudian *response* dari *request client* dikembalikan dalam format JSON.

2. *Suequence Diagram* Sistem



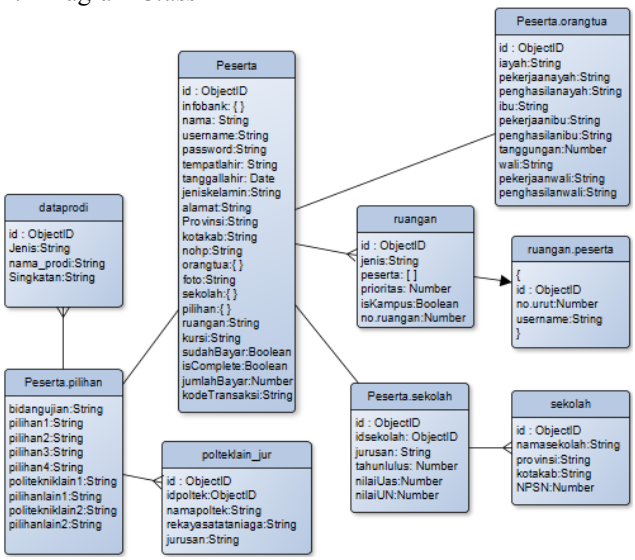
Gambar 5. *Suequence Diagram* Sistem

3. Diagram *Use Case*



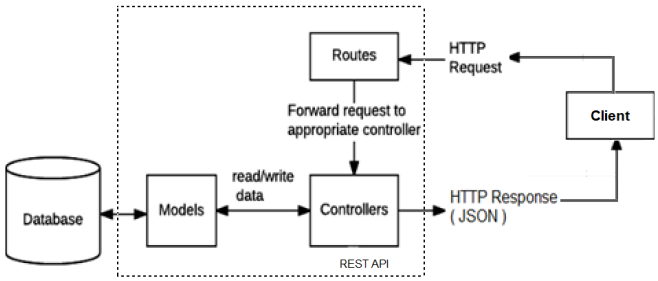
Gambar 4. Diagram *Use Case*

4. Diagram *Class*



Gambar 6. Diagram *Class*

5. *Pattern* Kode *Back-end*



Gambar 7. *Pattern* kode *back-end*

• *Model*: *model* mendefinisikan struktur *database* pada mongoDB dan bertugas untuk berinteraksi dengan *database*, serta menggunakan sebuah *script standalone* untuk menginisialisasikan beberapa pustaka *records*.

• *Controller*: *controller* berfungsi sebagai pengeksekusi *query*, yang kemudian berinteraksi dengan model untuk hubungan ke *database* nya. *Controller* juga berfungsi untuk mengambil data dari *request* yang dikirimkan ke api, baik melalui parameter ataupun *body* dari *http request*, dan mengembalikan respons sesuai dengan hasil *query* pada fungsi didalamnya dalam bentuk *Javascript Notation Object* (JSON).

• *Router*: *router* berfungsi sebagai *middleware*. *Router* mendefinisikan bagaimana respons URI terhadap *request client*. *Router* mendefinisikan URI dari tiap-tiap fungsi yang ada pada *controller* dan juga menentukan metode *request* yang digunakan, baik metode GET, POST, PUT, DELETE.

B. Pembangunan Sistem

Pada tahap ini, akan dilakukan pembangunan *backend server* dengan platform Node.js. Node.js berjalan di atas bahasa JavaScript, sehingga pembangunan program dilakukan menggunakan bahasa Javascript. Pemrograman dilakukan pada perangkat *Development*, kemudian dilakukan *deploy* ke perangkat server untuk kemudian dilakukan pengujian.

Server sendiri dibangun di atas sistem operasi linux, sehingga perlu dilakukan beberapa persiapan, antara lain:

1. Instalasi sistem operasi Linux pada komputer server.
2. Instalasi Node.js pada server.
3. Instalasi database MongoDB.
4. Menginstall dan menjalankan API yang telah dibuat pada perangkat development.

C. Pengujian dan Implementasi

Metode pengujian yang digunakan adalah black box testing yaitu langkah pengujian yang dilakukan untuk melihat hasil output dari aplikasi yang telah dibangun. Pengujian selanjutnya dilakukan uji pembebanan server atau load testing dengan mengirimkan sejumlah request secara bersamaan, untuk melihat response time dari server saat melayani jumlah virtual user yang bervariasi. Pengujian ini dilakukan di dua server, yakni server existing dan server yang dibangun untuk melihat perbandingan kinerja antar sistem.

IV. HASIL DAN PEMBAHASAN

A. Load Testing

Uji pembebanan pada server dilakukan dengan mengirimkan request secara bersamaan dengan jumlah virtual user yang bervariatif sesuai dengan perencanaan uji di subbab 3.2.6. Uji pembebanan dilakukan pada satu endpoint API dari sistem yang telah dibuat dan sistem existing dengan parameter request sebagai berikut:

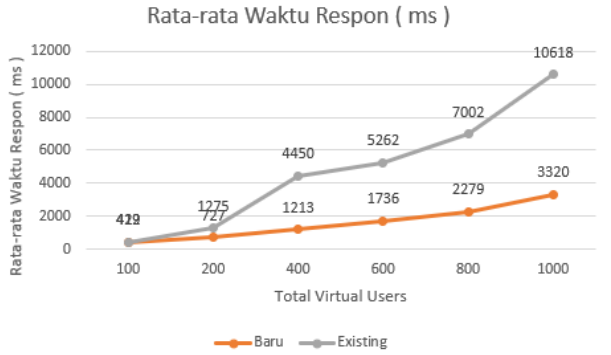
1. *Method*: POST
2. *Body request:* {"prioritas":"2","noruangan":"B32344", "jenis":"RKY","pesertaruangan":"18","isKampus":"false'}

Hasil pengujian dari skenario load testing antar kedua sistem tersebut terlampir pada tabel 4.1.

Tabel 3. Hasil pengujian load testing.

Total request	Sistem baru	Sistem *existing*
100	419	422
200	727	1275
400	1213	4450
600	1736	5262
800	2279	7002
1000	3320	10618

Dari hasil pengukuran pada tabel 3 maka dapat dihasilkan sebuah grafik yang dapat dilihat pada gambar 8



Gambar 8. Grafik rata-rata waktu respons *load testing.*

Berdasarkan hasil uji performa yang dapat dilihat pada gambar 8, secara umum terlihat bahwa rata-rata waktu respons meningkat sejalan dengan peningkatan jumlah *virtual user* yang melakukan *request* baik pada sistem yang telah dibuat maupun sistem existing dengan persentase peningkatan rata-rata waktu respons pada *back end* yang baru sebesar 52% dan persentase peningkatan rata-rata waktu respons pada *back end* sistem sebelumnya sebesar 111%.

Berdasarkan data dari kedua grafik tersebut, dapat dilihat bahwa terdapat perbedaan dengan data yang lebih besar menyebabkan peningkatan rata-rata waktu respons terhadap *request* sesuai dengan grafik yang terlihat pada gambar 4.1 dengan rata-rata waktu respons pada skenario *back end* yang baru sebesar lebih rendah dibandingkan dengan sistem *existing*.

Selisih rata-rata waktu respons dari sistem baru dan *existing* pada 100 *virtual users* sebesar 3 ms, pada 200 *virtual users* sebesar 548 ms, pada 400 *virtual users* sebesar 3237 ms, pada 600 *virtual users* sebesar 3526 ms, pada 800 *virtual users* sebesar 4723 ms, dan pada 1000 *virtual users* sebesar 7298 ms. Sehingga, berdasarkan data selisih rata-rata waktu respons meningkat seiring bertambahnya jumlah *virtual users* dengan rata-rata perbedaan pada keseluruhan data dari jumlah *virtual users* dari dua skenario uji sebesar 3222,5 ms lebih lambat pada sistem *existing*.

B. Black Box Testing

Metode Pengujian ini dilakukan untuk menguji keseluruhan fitur yang ada pada sistem apakah output dari endpoint API sesuai dengan yang diharapkan atau tidak.

Tabel 4. *black box testing modul login admin*

No	Fitur	Kondisi	Hasil
1	Login	Username dan password benar	Otentikasi berhasil
2	Login	Username dan password tidak sesuai	Otentikasi gagal, akses ditolak
3	Register staf	Membuat user staf dengan *username* dan *password* baru	User staf berhasil dibuat

Tabel 5. *black box testing modul ruangan*

No	Fitur	Kondisi	Hasil
1	Lihat metode acak ruangan	Input Jenis Ujian	Metode acak ruangan tertampil
2	Ubah metode acak ruangan	Input Jenis Ujian	Metode acak ruangan berubah
3	Buat ruangan	Input parameter ruangan	Ruangan berhasil dibuat
4	Statistik	-	Statistik ruangan dan peserta tertampil
5	Lihat ruangan-ruangan	Input jenis ujian	Ruangan-ruangan tertampil
6	Lihat ruangan	Input no. ruangan	Ruangan tertampil
7	Lihat semua ruangan	-	Semua ruangan tertampil

Tabel 6. black box testing modul pengecekan peserta

No	Fitur	Kondisi	Hasil
1	Hapus Peserta	Berjalan tiap jam 23.30, peserta yang belum bayar selama dua hari terhapus	Berjalan semestinya, peserta sesuai syarat terhapus
2	Ubah status bayar	Berjalan tiap 15 menit, cek semua peserta yang belum bayar ke API *payment gateway* (*dummy*)	Berjalan semestinya, peserta yang telah bayar diupdate statusnya.

Tabel 7. black box testing modul data statis

No	Fitur	Kondisi	Hasil
1	Prodi PNUP	-	Seluruh prodi di PNUP tertampil
2	Prodi PNUP	Input jenis ujian	Prodi di PNUP berdasrkan jenis ujian tertampil
3	Provinsi Indonesia	-	Provinsi di Indonesia tertampil

No	Fitur	Kondisi	Hasil
4	Kota/Kab	Input nama provinsi	Kota/kab di provinsi input tertampil
5	Sekolah	Input Kota/kab	Sekolah di kota/kab input tertampil
6	Cari sekolah	Input Kota/kab dan *keyword*	Sekolah di kota/kab input dan *keyword* tertampil
7	Politeknik lain	-	Daftar politeknik negeri lain tertampil
8	Prodi Politeknik lain	Input nama politeknik	Daftar prodi di politeknik input tertampil
9	Prodi Politeknik lain	Input nama politeknik, jenis ujian	Daftar prodi di politeknik input dan filter jenis ujian tertampil
10	Jurusan sekolah	-	Jurusan sma/k sederajat dan jenis tesnya tertampil
	Jurusan SMA	-	jurusan-jurusan di SMA dan jenis tes yang bisa diikuti tertampil
12	Jurusan SMK	Input Teknik / NonTeknik	jurusan-jurusan di SMK dengan filter dan jenis tes yang bisa diikuti tertampil

Tabel 8. *black box testing* modul staf

No	Fitur	Kondisi	Hasil
1	Login Staf	Username dan password benar	Otentikasi berhasil
2	Login Staf	Username dan password tidak sesuai	Otentikasi gagal, akses ditolak
3	Pendaftaran on the spot	Memasukkan data peserta	Registrasi berhasil, akun peserta dibuat
4	Cek tahun ijazah	Tahun ijazah <=3 tahun terakhir	Tahun ijazah ditolak

Tabel 9. *black box testing* modul registrasi awal peserta

No	Fitur	Kondisi	Hasil
1	Autentikasi	Username dan password benar, telah membayar	Otentikasi berhasil
2	Autentikasi	Username dan password tidak sesuai	Otentikasi gagal, akses ditolak
3	Autentikasi	Peserta belum membayar	Otentikasi gagal, akses ditolak
4	Cek tahun ijazah	Tahun ijazah <=3 tahun terakhir	Tahun ijazah ditolak
5	Cek nomor ponsel	Nomor ponsel belum digunakan	Nomor ponsel diizinkan
6	Cek nomor ponsel	Nomor ponsel telah digunakan	Nomor ponsel ditolak
7	Cek nomor ponsel	Format nomor ponsel salah	Nomor ponsel ditolak
8	Registrasi *online*	Input data peserta	User,pass, kode bayar didapatkan

Tabel 9. *black box testing* modul registrasi lanjutan

No	Fitur	Kondisi	Hasil
1	*Upload* foto	Jenis jpg/png, ukuran <1MB	Upload berhasil
2	*Upload* foto	Jenis selain jpg/png	Upload ditolak
3	*Upload* foto	ukuran >1MB	Upload ditolak
4	Lengkapi data	Melengkapi detail data peserta	Registrasi berhasil
5	*Attempt* kursi & cetak kartu tes	*Iinput username,* jenis ujian	Data kartu tes beserta ruangan dan kursi ditampilkan

V. KESIMPULAN

Setelah melakukan pengembangan, pengujian, dan implementasi sistem UMPN di PNUP dapat disimpulkan sebagai berikut:

1. Penelitian ini berhasil membangun *back-end server* untuk sistem ujian masuk politeknik negeri (UMPN) Politeknik Negeri Ujung Pandang menggunakan arsitektur REST dan platform Node.js dengan kinerja dan fungsionalitas yang baik.
2. Rata-rata waktu respons server meningkat seiring jumlah *virtual user* yang mengakses sistem secara bersamaan diperbesar dengan rata-rata peningkatan sebesar 52% pada skenario yang diujikan.
3. Sistem yang dibuat memiliki kinerja yang lebih baik dibandingkan sistem *existing* berdasarkan hasil uji dengan rata-rata selisih waktu respons berdasarkan skenario pengujian sebesar 3222,5 ms

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Allah *Subhanahu Wa Ta'Ala*, kedua orang tua, keluarga, kedua dosen pembimbing, seluruh dosen Teknik Elektro khusunya program studi D4 Teknik Komputer dan Jaringan dan teman-teman se-program studi TKJ angkatan 2016.

REFERENSI

[1] S. Kosasi, "Perancangan Dan Pemanfaatan E-Commerce Untuk Memperluas Pasar Produk Furniture," *Semin. Nas. Teknol. Inf. dan Komun. 2015 (SENTIKA 2015) Yogyakarta, 28 Maret 2015 ISSN 2089-9815 Peranc.*, vol. 2015, no. Sentika, pp. 17–24, 2015.

[2] U. Rahardja, Q. Aini, and N. P. L. Santoso, "Pengintegrasian YII Framework Berbasis API pada Sistem Penilaian Absensi," *Sisfotenika*, vol. 8, no. 2, p. 140, 2018.

[3] I. A. Faruqi, S. F. S. Gumilang, and M. A. Hasibuan, "Perancangan Back-End Aplikasi Rumantara Dengan Gaya Arsitektur Rest Menggunakan Metode Iterative Incremental," *eProceedings Eng.*, vol. 5, no. 1, pp. 1411–1417, 2018.

[4] I. B. P. Manuaba and E. Rudiastini, "API REST Web service and backend system of Lecturer's Assessment Information System on Politeknik

Negeri Bali," *J. Phys. Conf. Ser.*, vol. 953, no. 1, 2018.

[5] A. C. Rompis and R. F. Aji, "Perbandingan Performa Kinerja Node.js, PHP, dan Python dalam Aplikasi REST," *CogITo Smart J.*, vol. 4, no. 1, p. 171, 2018.

[6] S. Bangare, S. Gupta, M. Dalal, and A. Inamdar, "Using Node.Js to Build High Speed and Scalable Backend Database Server," i*nternational J. Res. Advent Technol. (E-ISSN 2321-9637)*, vol. 4, no. May, p. 19, 2016.

[7] M. Luqman, "KEAMANAN PERANGKAT LUNAK PADA BAHASA PEMROGRAMAN NODE.JS UNTUK APLIKASI BERBASIS WEB," 2016. Tidak Dipublikasikan.

[8] S. Syam, "Wawancara Performansi Sistem UMPN PNUP," Makassar, 2019. Tidak Dipublikasikan.

[9] A. Angreini, I. K. Yusri, E. Tungadi, and M. N. Y. Utomo, "Sistem Pendaftaran Ujian Masuk Politeknik Negeri Ujung Pandang," 2019. Tidak Dipublikasikan.

[10] M. Y. Utomo, "Pengembangan Aplikasi Web Pendaftaran Online UMPN," Makassar, 2014. Tidak Dipublikasikan.

[11] Feridi, "Mengenal RESTful Web Services," *Codepolitan.com*, 2019. [Online]. Tersedia di: https://www.codepolitan.com/mengenal-restful-web-services. [Diakses: 05-Agustus-2019].

[12] Sinaryuda, "Pengenalan MongoDB," *sinaryuda.web.id*, 2010. [Online]. Tersedia di: https://www.sinaryuda.web.id/tutorial/pengenalan-mongodb.html. [Diakses: 01-Agustus-2019].