Jurnal Minfo Polgan
Volume 12, Nomor 1, Juli 2023
DOI : https://doi.org/10.33395/jmp.v12i1.12691
e-ISSN : 2797-3298
p-ISSN : 2089-9424

Terbit : 01 Juli 2023

Implementasi dan Pengujian REST API Sistem Reservasi Ruang Rapat dengan Metode *Black Box Testing*

(Studi Kasus: PT Lizzie Parra Kreasi)

[1]Rangga Gelar Guntara, [2] Varinia Azkarin
[1, 2]Universitas Pendidikan Indonesia

[1]ranggagelar@upi.edu, [2]variniaazkarin@upi.edu

ABSTRAK

Pengembangan *backend* merupakan salah satu aspek yang penting dalam mengembangkan *web*. Untuk menghubungkan interaksi antara *browser client* dan *database*, diperlukan *application programming interface* (API). Salah satu arsitektur yang umum digunakan dalam pengembangan *backend* adalah *representational state transfer* (REST). Untuk membangun REST API, digunakan *platform* Node.js dalam menjalankan Javascript pada *backend server*. REST API dan *database management system* untuk sistem reservasi PT Lizzie Parra Kreasi dirancang dan diimplementasikan dengan tujuan menunjang aplikasi *client* yang telah ada, agar proses manajemen data reservasi lebih efisien. Tahapan yang dilalui dalam penelitian ini meliputi identifikasi masalah, pengumpulan data dan studi literatur, perancangan, implementasi, hingga pengujian. Proses perancangan dan implementasi REST API berhasil dijalankan dengan baik. Adapun pengujian yang dilakukan dengan *black box testing*, menunjukkan bahwa semua fungsi setiap *endpoint* API berjalan sesuai dengan *output* yang diharapkan. Seluruh *endpoint* berjalan sesuai dengan *output* dan kode respons yang diharapkan, baik dengan metode HTTP GET, POST, PUT, ataupun DELETE.

Kata Kunci: *Web*, *Backend*, REST API, Node.js, PT Lizzie Parra Kreasi

PENDAHULUAN

Web service dan aplikasi berbasis *web* merupakan salah satu teknologi yang banyak dikembangkan untuk membantu aktivitas manusia di tengah perkembangan teknologi yang pesat. Saat ini, banyak aplikasi *web* yang dalam pengembangannya tidak luput dari dua bagian yang penting, yaitu *frontend* dan *backend*. *Frontend* merupakan bagian *web* yang berinteraksi langsung dengan pengguna. Sedangkan *backend* adalah bagian *web* yang biasanya berkaitan dengan tiga hal, yaitu: *server*, aplikasi, dan basis data (Abdullah & Zeki, 2014).

Pengembangan *backend* berfokus pada aspek-aspek yang berada di sisi *server*. Kode program pada aplikasi *backend* memungkinkan interaksi antara *browser* dan memperoleh data dari *database*. Agar sebuah *server* dapat berinteraksi dengan *platform client* yang berbeda-beda, dibutuhkan teknologi yang dikenal sebagai *application programming interface* (API). API merupakan sekumpulan instruksi dalam sebuah perangkat lunak, yang berfungsi sebagai penghubung antara aplikasi dengan sumber daya lain seperti *server* atau *database* (Guntara, 2023).

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Jurnal Minfo Polgan
Volume 12, Nomor 1, Juli 2023 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v12i1.12691 p-ISSN : 2089-9424

Dalam pengembangan website, *representational state transfer* (REST) merupakan salah satu arsitektur API yang dibuat pada web server, untuk melakukan transfer data melalui protokol HTTP (Suzanti, dkk., 2020). API dengan arsitektur REST dapat dibuat dengan bahasa pemrograman seperti PHP, Ruby, ataupun Javascript dengan *runtime* Node.js. Dalam penelitian yang dilakukan oleh Prayogi, dkk. (2020), implementasi REST API menggunakan Node.js memiliki performa yang lebih baik dibandingkan dengan implementasi REST API yang menggunakan bahasa pemrograman PHP.

Node.js merupakan sebuah *platform* yang berjalan pada browser untuk membaca bahasa pemrograman javascript sebagai pemrograman *backend*. Mesin ini memiliki kelebihan tersendiri pada sistem *non-blocking*, yang memungkinkan operasi atau program dijalankan secara bersamaan, sehingga memungkinkan untuk menyelesaikan banyak request secara serentak atau paralel (Mubariz, dkk., 2020). Dalam penelitian yang dilakukan oleh Rompis & Aji (2018) Node.js menjadi bahasa pemrograman yang paling cocok dalam menyediakan respon secara langsung terhadap *request* yang masuk.

PT Lizzie Parra Kreasi saat ini telah memiliki aplikasi sistem reservasi ruang rapat berbasis web. Akan tetapi, aplikasi yang tersedia saat ini belum memiliki aplikasi *backend* dan *database management system*. Keterbatasan ini dapat menyebabkan manajemen sistem reservasi kurang efisien, karena tidak dapat menyimpan riwayat reservasi. Menurut Feng (2020), sistem yang memiliki sistem basis data lebih efektif dalam integrasi data, dan memberikan kemudahan dalam meningkatkan efisiensi manajemen informasi.

Oleh karena itu, diperlukan implementasi REST API untuk membangun sistem reservasi yang lebih efisien. Dalam penelitian ini, akan membahas mengenai bagaimana rancangan dan implementasi REST API pada sistem reservasi ruang rapat PT Lizzie Parra Kreasi. Selanjutnya, hasil implementasi tersebut akan diuji menggunakan metode *black box,* yaitu metode yang berfokus pada fungsional aplikasi, untuk memastikan apakah *input* dan *output* yang diharapkan telah sesuai.

TINJAUAN PUSTAKA

Pengembangan Web Secara *Backend*

Backend adalah salah satu komponen penting dalam *website.* Rahaman, dkk., (2022) mengemukakan bahwa *backend* adalah kumpulan proses yang terjadi di belakang layar *website* ketika pengguna melakukan permintaan kepada *server*. Pengembangan *backend* dengan bahasa pemrograman seperti PHP, dan NodeJS dikenal sebagai *server-side programming*.

Application Programming Interface (API)

Menurut Guntara (2023) *application programming interface* (API) merupakan sekumpulan instruksi yang dipakai oleh perangkat lunak untuk berinteraksi antara perangkat lunak. API memiliki kemampuan untuk menghubungkan aplikasi ke aplikasi lain, memungkinkan *programmer* menggunakan dan mengakses fitur yang tersedia pada sistem serta sumber daya lainnya seperti *server*, basis data, ataupun perangkat keras. Salah satu arsitektur dari API adalah arsitektur REST.

API berarsitektur REST memungkinkan sistem-sistem berinteraksi satu sama lain, dan memudahkan pengiriman dan penerimaaan data. Dalam arsitektur ini, sumber daya database dapat dihubungkan dengan endpoint pada REST API. Cara kerja REST dilakukan dengan navigasi melalui link HTTP dalam aktivitas tertentu, yang memungkinkan perpindahan *state*. Setiap *call* API yang dilakukan dalam arsitektur ini dilakukan melalui protokol HTTP dengan beberapa metode yang umum digunakan seperti GET, POST, PUT, dan DELETE (Choirudin & Adil, 2019).

Node JS

Untuk mendukung Javascript sebagai berjalan dalam *server*, terdapat sebuah perangkat lunak yang dikenal dengan nama Node.js. Perangkat lunak tersebut dibuat untuk mengembangkan aplikasi berbasis web, yang ditulis dalam bahasa pemrograman Javascript, dan memiliki *library* HTTP tersendiri sehingga tidak perlu menjalankan aplikasi pendukung web seperti Apache (Huda, dkk., 2020).

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Jurnal Minfo Polgan
Volume 12, Nomor 1, Juli 2023 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v12i1.12691 p-ISSN : 2089-9424

Node JS memiliki keunggulan yaitu penggunaan teknologi *non-blocking*, yang memungkinkan eksekusi tugas sekaligus tanpa harus menunggu tugas lain terselesaikan, sehingga performanya sangat responsif dan efisien (Huang, 2020).

Object Relational Mapping (ORM)

Menurut Bauer (dalam Rahmat, dkk., 2023), *object relational mapping* (ORM) adalah suatu teknik yang memungkinkan pengguna melakukan penyimpanan objek ke dalam tabel database secara otomatis. Metode ORM menggunakan metadata yang menunjukkan relasi antara objek dan database. Dengan demikian, tabel RDBMS dapat dipetakan ke dalam *class* dalam *object oriented programming*. Salah satu ORM populer yang mendukung Node JS adalah sequelize.

Javascript Object Notation (JSON)

Javascript Object Notation (JSON) adalah suatu format data yang berbasis tipe data dari bahasa pemrograman Javascript (Pezoa, dkk., 2016). JSON telah menjadi format data populer, dan dalam beberapa tahun telah menjadi format pertukaran data di *world wide web* (Lv, dkk., 2018). Dalam penulisannya, sintaks JSON memiliki dua struktur yang terdiri dari objek dan *array*, Setiap objek dalam JSON diawali dengan kurung kurawal buka, dan diakhiri dengan kurung kurawal tutup. Setiap string memiliki tanda titik dua dan memiliki *value*, dan setiap *string* dipisahkan oleh tanda koma (Rosid, 2017). Berikut ini merupakan contoh dari format JSON.

```
{
    "username": "user1",
    "full_name": "User Satu",
    "email": "user1@gmail.com",
    "password": "user123",
    "job_title": "Developer Intern",
    "division_id": 1,
    "joblevel_id": 1
}
```

Gambar 1. Contoh Format JSON

Pengujian *Black Box*

Metode *blackbox* merupakan metode pengujian yang memungkinkan pengujian fungsional setiap fitur aplikasi dalam pengujian sebuah perangkat lunak. Metode ini hanya berfokus pada pengujian fungsionalitas perangkat lunak, memungkinkan penguji untuk menguji fungsionalitas melalui berbagai input tertentu pada setiap *form* untuk mengetahui apakah program yang dibuat telah memenuhi spesifikasi yang direncanakan (Shadiq, dkk., 2021).

METODE PENELITIAN

Penelitian ini dilakukan di PT Lizzie Parra Kreasi, yang berlokasi di Kebayoran Baru, Jakarta Selatan. Alur penelitian yang dilakukan terdiri dari identifikasi masalah, studi literatur, perancangan dan implementasi API, pengujian menggunakan metode *black box*, serta pengambilan kesimpulan penelitian.

Identifikasi Masalah

Tahapan penelitian diawali dengan mengidentifikasi masalah, yaitu belum adanya aplikasi backend serta *database management system* yang memungkinkan proses manajemen data. Oleh karena itu, diperlukan sebuah REST API sebagai perantara frontend dan server dalam proses pertukaran data.

Pengumpulan Data dan Studi Literatur

Tahapan ini dilakukan untuk mengkaji teori-teori yang relevan dengan penelitian. Selain itu, pengumpulan informasi atau data juga dilakukan melalui observasi terhadap PT Lizzie Parra Kreasi.

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Jurnal Minfo Polgan
Volume 12, Nomor 1, Juli 2023 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v12i1.12691 p-ISSN : 2089-9424

Perancangan dan Implementasi

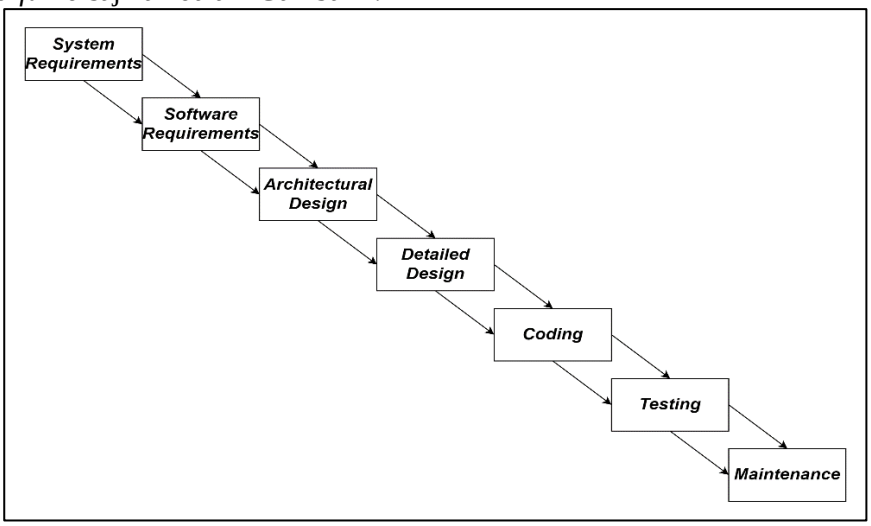
Tahapan ini membahas mengenai bagaimana perancangan REST API yang akan dibangun dalam sistem reservasi ruang rapat PT Lizzie Parra Kreasi. Langkah ini dilakukan dengan melakukan perancangan per *endpoint* seperti URI dan parameter yang digunakan, lalu dilakukan implementasi call API menggunakan aplikasi Postman. Proses perancangan dan implementasi ini menggunakan protokol HTTP dengan metode GET untuk memperoleh data, metode POST untuk mengirimkan data, PUT untuk melakukan perubahan data, serta DELETE untuk menghapus data dari *database*.

Pengujian

Setelah melakukan perancangan dan implementasi, dilakukan tahapan pengujian dengan metode *black box.* Pengujian *black box* berfokus pada fungsionalitas aplikasi. Tahapan ini dilakukan guna mengetahui bagaimana kesesuaian antara *input* yang dimasukkan serta *output* yang diharapkan.

Metode Pengembangan Perangkat Lunak

Metode pengembangan *software* juga disebut dengan istilah *Software Development Life Cycle*. Dalam penelitian ini, penulis menggunakan SDLC metode *waterfall.* Menurut (Rastogi, 2015), secara umum terdapat beberapa tahapan dalam metode *waterfall*, diantaranya adalah: (1) perencanaan dan visualisasi; (2) analisis kebutuhan; (3) pemodelan dan desain perangkat lunak; (4) pengkodean; (5) dokumentasi; (6) pengujian; dan (7) *deployment* dan pemeliharaan. Gambaran alur metode *waterfall* disajikan dalam Gambar 2.



Gambar 2. SDLC Metode *Waterfall* (Rastogi, 2015)

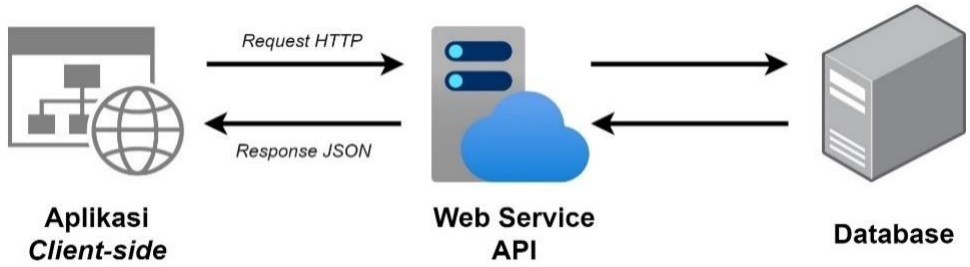
HASIL DAN PEMBAHASAN

Perancangan Arsitektur Sistem

Tahapan ini bertujuan untuk memberi gambaran umum mengenai cara kerja sistem, yang ditunjukkan pada Gambar 2. Terdapat tiga komponen utama dalam arsitektur ini, yaitu: aplikasi HRIS *client-side* yang diakses oleh pengguna melalui browser, Web Service API, dan Database sebagai tempat penyimpanan data. Dalam setiap proses *request*, pengguna akan mengirimkan *request* dari aplikasi *client-side* melalui protokol HTTP dengan metode GET, POST, PUT, ataupun DELETE ke Web Service API. Melalui program yang telah dibuat, *request* tersebut akan diproses dalam API untuk diteruskan ke Database. Proses tersebut dapat berupa pengambilan data dari Database, pemasukan data, atau modifikasi data. API akan memberi *response* berupa JSON yang berisi pesan apakah data sudah terkirim, ataupun memberi pesan *error* jika terjadi kesalahan dalam proses *request.*

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

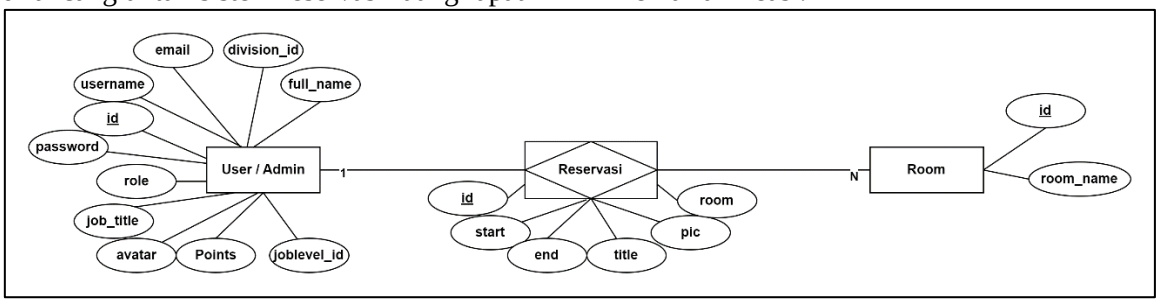
Jurnal Minfo Polgan
Volume 12, Nomor 1, Juli 2023
DOI : https://doi.org/10.33395/jmp.v12i1.12691
e-ISSN : 2797-3298
p-ISSN : 2089-9424



Gambar 3. Arsitektur Sistem

Perancangan *Database*

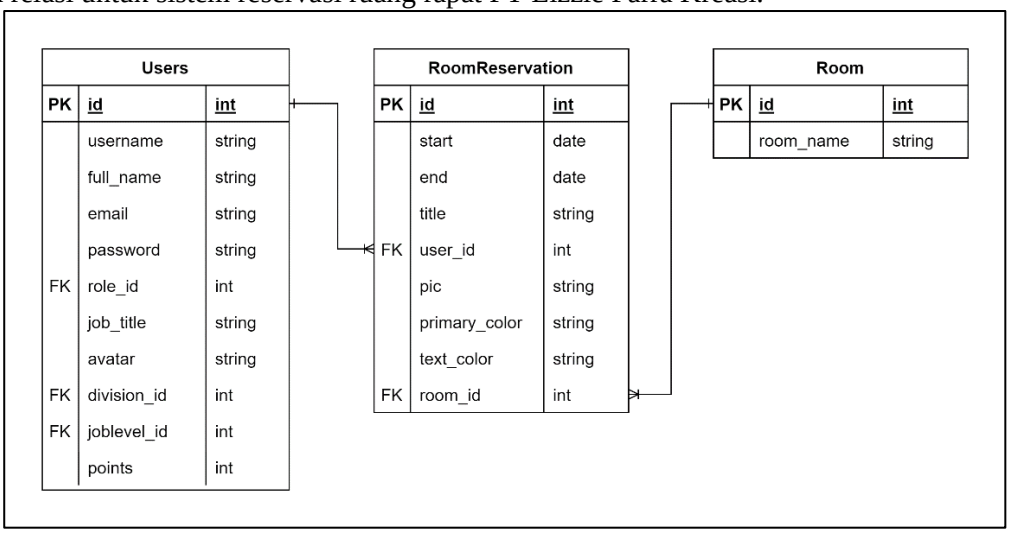
Entity Relationship Diagram (ERD) merupakan suatu diagram terstruktur yang digunakan dalam tahap perancangan database (Larassati, dkk., 2019). Berikut ini merupakan ERD yang dirancang untuk sistem reservasi ruang rapat PT Lizzie Parra Kreasi.



Gambar 4. ERD Sistem Reservasi Ruang Rapat

Implementasi Database

Untuk menggambarkan bagaimana skema relasi *database*, digunakan notasi *crow's foot*. Dalam notasi ini, entitas digambarkan dengan sebuah kotak (tabel), dengan atribut yang berjajar di bawah nama entitas, kemudian dikoneksikan menggunakan garis relationship yang menyerupai kaki burung untuk menandai jenis relasi entitas tersebut (Puja, dkk., 2019). Berikut ini merupakan skema relasi untuk sistem reservasi ruang rapat PT Lizzie Parra Kreasi.



Gambar 5. Skema Relasi Sistem Reservasi Ruang Rapat

Implementasi *Endpoint*

Berikut ini merupakan daftar *endpoint* yang digunakan untuk implementasi API sistem reservasi PT Lizzie Parra Kreasi.

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

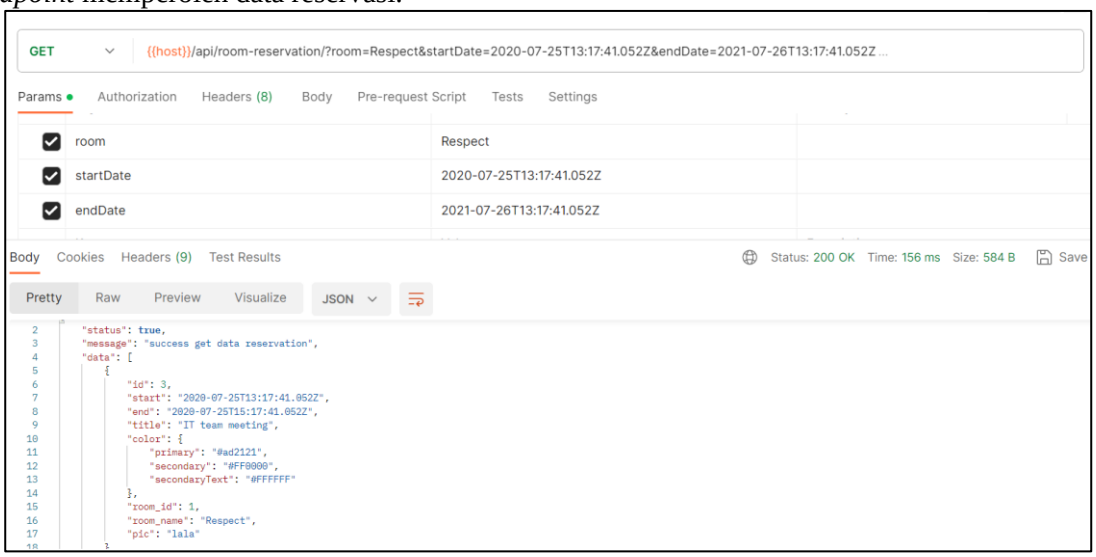
Jurnal Minfo Polgan
Volume 12, Nomor 1, Juli 2023 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v12i1.12691 p-ISSN : 2089-9424

Tabel 1. Daftar *Endpoint* API

No.	Metode HTTP	URI	Parameter	Keterangan
1.	GET	/api/room-reservation	room, startDate, endDate (*query string*)	Memperoleh Data Reservasi
2.	GET	/api/room-reservation/:id	ID reservasi	Memperoleh Data Reservasi Berdasarkan ID
3.	POST	/api/room-reservation	-	Membuat Data Reservasi
4.	PUT	/api/room-reservation/:id	ID reservasi	Memperbaharui Data Reservasi
5.	DELETE	/api/room-reservation/:id	ID reservasi	Menghapus Data Reservasi

1. Memperoleh Data Reservasi

Endpoint ini ditujukan untuk mendapatkan data reservasi, berdasarkan *query string* yang dimasukkan oleh *user*. Data yang didapatkan dari *endpoint* ini diperoleh berdasarkan rentang waktu sesuai dengan variabel startDate dan endDate, serta variabel room jika dibutuhkan. *Role* yang dapat mengakses *endpoint* ini adalah Admin dan User. Berikut ini merupakan hasil implementasi dari *endpoint* memperoleh data reservasi.



Gambar 6. Hasil *Call* API Memperoleh Data Reservasi

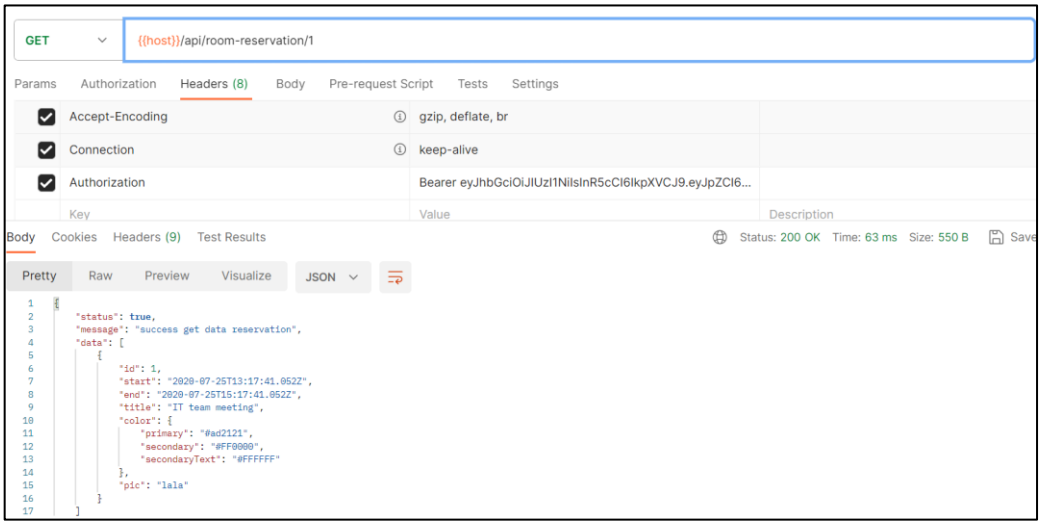
Gambar 2 menunjukkan hasil implementasi dari *call* API endpoint ini. Dalam implementasi tersebut, data yang dimasukkan berupa *query string* room yang diisi dengan nama ruang rapat, startDate yang diisi dengan rentang tanggal awal, dan endDate yang diisi dengan rentang tanggal akhir. Hasil yang dikirimkan dari API merupakan *response* JSON berdasarkan *request* yang dieksekusi.

2. Memperoleh Data Reservasi Berdasarkan ID

Endpoint ini ditujukan untuk memperoleh data reservasi ruang rapat berdasarkan ID-nya. Data yang diperoleh dari *endpoint* ini didapatkan sesuai dengan pengisian parameter ID pada URI yang dimasukkan oleh *user*. *Role* yang dapat mengakses *endpoint* ini adalah Admin dan User. Berikut ini merupakan hasil implementasi dari *endpoint* tersebut.

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Jurnal Minfo Polgan
Volume 12, Nomor 1, Juli 2023
DOI : https://doi.org/10.33395/jmp.v12i1.12691
e-ISSN : 2797-3298
p-ISSN : 2089-9424

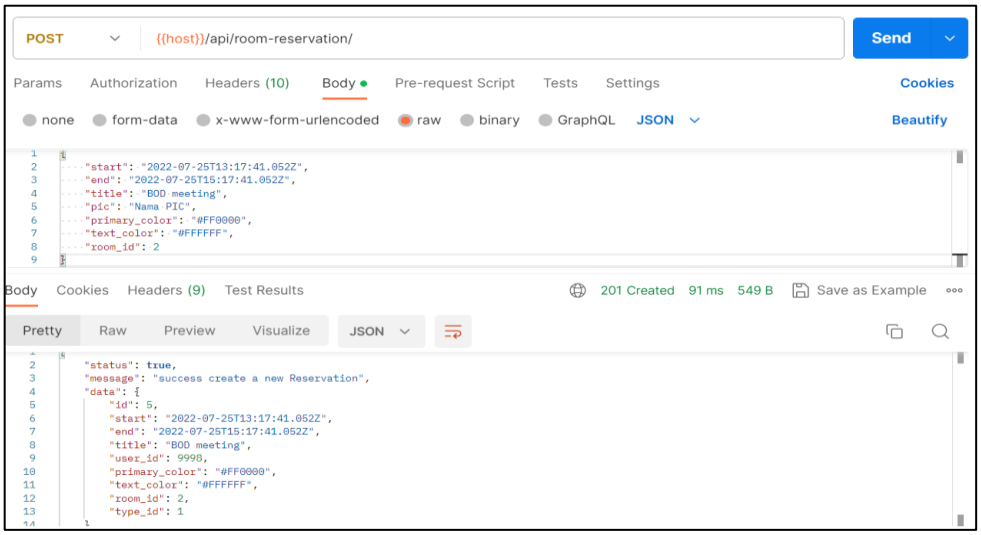


Gambar 7. Hasil *Call* API Memperoleh Data Reservasi Berdasarkan ID

Gambar 3 menunjukkan hasil dari *call* API *endpoint* ini. Dalam implementasi tersebut, *user* memasukkan ID data reservasi ruang rapat yang diperoleh. Kemudian, API akan mengirimkan data-data reservasi ruang rapat yang ID-nya sesuai dengan parameter URI yang dikirimkan.

3. Membuat Data Reservasi Baru

Endpoint ini ditujukan untuk membuat data reservasi ruang rapat yang baru. Untuk melakukan *call* API ini, *user* perlu mengirimkan *request body* JSON yang berisi *field* beserta *value*-nya. Data yang dibutuhkan dalam *request body* diantaranya adalah: (1) start yang diisi dengan waktu mulai rapat; (2) end yang diisi dengan waktu akhir rapat; (3) title yang diisi dengan judul rapat; (4) pic yang diisi dengan nama penanggungjawab rapat; (4) primary_color yang diisi dengan kode *hex* warna kolom reservasi; (5) text_color yang diisi dengan kode *hex* warna teks reservasi; dan (6) room_id yang diisi dengan ID ruang rapat. *Role* yang dapat mengakses *endpoint* ini adalah Admin dan User.



Gambar 8. Hasil *Call* API Membuat Data Reservasi Baru

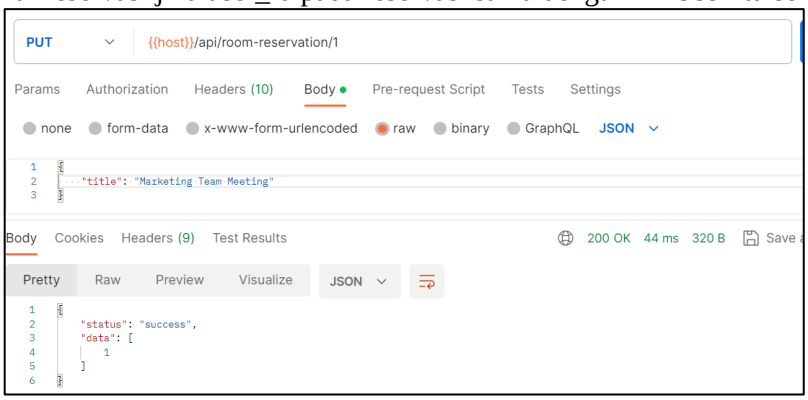
Dalam hasil *call* API pada Gambar 4, *user* mengirimkan data reservasi baru dengan mengisi *field* sesuai tipe datanya masing-masing. Ketika mengirimkan *request*, API akan mengirimkan *response* berupa kode respon, pesan, serta data lengkap dari reservasi yang sudah terbuat.

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Jurnal Minfo Polgan
Volume 12, Nomor 1, Juli 2023 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v12i1.12691 p-ISSN : 2089-9424

4. Memperbaharui Data Reservasi

Endpoint ini ditujukan untuk memperbaharui data reservasi yang ada. Untuk memperbaharui data, *user* perlu memasukkan ID reservasi serta *request body* berisi *field* yang akan diperbaharui. *Role* Admin dalam *endpoint* ini dapat memperbaharui semua reservasi. Sedangkan *role* User hanya bisa memperbaharui reservasi jika user_id pada reservasi sama dengan ID User itu sendiri.

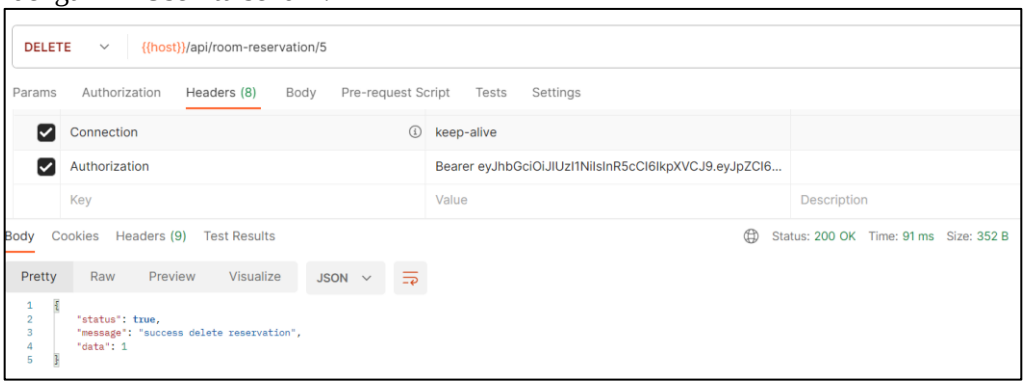


Gambar 9. Hasil *Call* API Memperbaharui Data Reservasi

Dalam hasil *call* API yang ditunjukkan pada Gambar 5, *request body* diisi dengan *field* yang akan diubah, misalnya judul reservasi. *Field* yang diisi pada *request body* bisa diisi lebih dari satu. Pada parameter URI, diisi dengan ID reservasi yang akan diperbaharui. Kemudian, API akan memberikan kode respons, pesan respons, serta jumlah *row* yang telah diperbaharui.

5. Menghapus Data Reservasi

Endpoint ini ditujukan untuk menghapus data reservasi. Untuk menghapus data, *user* perlu memasukkan ID reservasi yang akan dihapus. *Role* Admin dalam *endpoint* ini dapat menghapus semua reservasi, sedangkan *role* User hanya bisa menghapus reservasi jika user_id pada reservasi sama dengan ID User itu sendiri.



Gambar 10. Hasil *Call* API Menghapus Data Reservasi

Dalam hasil *call* API yang ditunjukkan pada Gambar 6, parameter URI diisi dengan ID reservasi yang akan dihapus. Setelah dilakukan *request*, maka API akan mengirimkan kode respons, pesan respons, serta jumlah *row* data reservasi yang telah dihapus.

Pengujian

Setelah tahapan implementasi, selanjutnya adalah melakukan pengujian pada API dengan metode *black box*. Tahapan ini bertujuan untuk mengetahui apakah fungsional dari aplikasi telah berjalan dengan baik dan sesuai dengan *output* yang diharapkan. Pengujian dilakukan pada masing-masing *endpoint*, dengan hasil yang disajikan pada tabel berikut.

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Jurnal Minfo Polgan
Volume 12, Nomor 1, Juli 2023 e-ISSN : 2797-3298
DOI : https://doi.org/10.33395/jmp.v12i1.12691 p-ISSN : 2089-9424

Tabel 2. Hasil Pengujian *Black Box*

No.	Kegiatan Pengujian	*Test Case*	*Output* yang Diharapkan	Hasil	Kesimpulan
1.	Membuat data reservasi baru	Memasukkan value pada key JSON start, end, title, pic, primary_color, text_color, dan room_id	Muncul pesan sukses dan data reservasi ruang rapat yang telah dibuat.	Sesuai	Valid
2.	Memperoleh data reservasi	Memasukkan query startDate, query endDate, query room	Muncul seluruh data reservasi berdasarkan dengan *input query* yang dimasukkan.	Sesuai	Valid
3.	Memperoleh data reservasi berdasarkan ID	Memasuki URI /api/room-reservation/:id dengan parameter ID reservasi	Muncul data reservasi sesuai ID yang dimasukkan	Sesuai	Valid
4.	Memperbaharui data reservasi	Memasukkan value pada key JSON start, end, title, pic, primary_color, text_color, atau room_id	Muncul pesan sukses dan data reservasi baru yang dimasukkan	Sesuai	Valid
5.	Menghapus data reservasi	Memasuki URI /api/room-reservation/:id dengan parameter ID reservasi	Muncul pesan sukses	Sesuai	Valid

KESIMPULAN

Berdasarkan hasil penelitian, dapat disimpulkan bahwa REST API sistem reservasi ruang rapat PT Lizzie Parra Kreasi dapat dirancang dan diimplementasikan dengan baik. Tahapan yang dilalui dalam penelitian ini meliputi identifikasi masalah, pengumpulan data dan studi literatur, perancangan, implementasi, hingga pengujian dengan metode *black box*. Penggunaan API pada penelitian ini memungkinkan aplikasi *clien*t sistem reservasi dapat melakukan pertukaran data dengan *database*, sehingga riwayat reservasi dapat tersimpan dengan baik.

Adapun hasil pengujian dengan metode *black box testing*, yang menunjukkan bahwa semua fungsi setiap *endpoint* API berjalan sesuai dengan *output* yang diharapkan. Melalui metode HTTP, seluruh *endpoint* dapat berjalan dengan baik, mulai dari perolehan data dengan metode GET, pembaharuan data dengan metode PUT, penambahan data dengan metode POST, dan penghapusan data dengan metode DELETE.

REFERENSI

Abdullah, H. M., & Zeki, A. M. (2014). Frontend and backend web technologies in social networking sites: Facebook as an example. *Proceedings - 3rd International Conference on Advanced Computer Science Applications and Technologies, ACSAT 2014*, 85–89. https://doi.org/10.1109/ACSAT.2014.22

Choirudin, R., & Adil, A. (2019). Implementasi Rest Api Web Service dalam Membangun Aplikasi Multiplatform untuk Usaha Jasa. *MATRIK : Jurnal Manajemen, Teknik Informatika Dan Rekayasa Komputer*, *18*(2), 284–293. https://doi.org/10.30812/matrik.v18i2.407

Feng, L. (2020). On the Application of Computer Database System In Information Management. *2020 Conference on Educational Science and Educational Skills*, 246–250. https://doi.org/10.38007/proceedings.0000595

Guntara, R. G. (2023). Aplikasi Deteksi Phising Berbasis Android Menggunakan Metode

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Pengembangan Perangkat Lunak DSRM. *Jurnal Minfo Polgan*, *12*(1), 303–310. https://doi.org/https://doi.org/10.33395/jmp.v12i1.12379 e-ISSN

Huang, X. (2020). *Research and Application of Node . js Core Technology*. 29–32.

Huda, D. N., Saputra, A., & Yulinda. (2020). Perancangan Aplikasi It Help Desk Menggunakan Platform Node.Js Pada Mittasys. *Jurnal Bangkit Indonesia*, *9*(1), 137–143. https://doi.org/10.52771/bangkitindonesia.v9i1.144

Larassati, M., Latukolan, A., Arwan, A., & Ananta, M. T. (2019). Pengembangan Sistem Pemetaan Otomatis Entity Relationship Diagram Ke Dalam Database. *Urnal Pengembangan Teknologi Informasi Dan Ilmu Komputer* , *3*(4), 4059. http://j-ptiik.ub.ac.id

Lv, T., Yan, P., & He, W. (2018). Survey on JSON Data Modelling. *Journal of Physics: Conference Series*, *1069*(1). https://doi.org/10.1088/1742-6596/1069/1/012101

Mubariz, A., Nur, D., Tungadi, E., & Utomo, M. N. Y. (2020). Perancangan Back-End Server Menggunakan Arsitektur Rest dan Platform Node . JS (Studi Kasus : Sistem Pendaftaran Ujian Masuk Politeknik Negeri Ujung Pandang). *Seminar Nasional Teknik Elektro Dan Informatika (SNTEI)*, 72–77.

Pezoa, F., Reutter, J. L., Suarez, F., Ugarte, M., & Vrgoč, D. (2016). Foundations of JSON schema. *25th International World Wide Web Conference, WWW 2016*, 263–273. https://doi.org/10.1145/2872427.2883029

Prayogi, A. A., Niswar, M., Indrabayu, & Rijal, M. (2020). Design and Implementation of REST API for Academic Information System. *IOP Conference Series: Materials Science and Engineering*, *875*(1). https://doi.org/10.1088/1757-899X/875/1/012047

Puja, I., Poscic, P., & Jaksic, D. (2019). Overview and comparison of several relational database modelling metodologies and notations. *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2019 - Proceedings*, 1641–1646. https://doi.org/10.23919/MIPRO.2019.8756667

Rahaman, A., Gayatri, V., Kiran, C. S., Pavan, K. S., Bhumika, B., & Sateesh, V. (2022). Development of Web Applications by Integrating Frontend and Backend Tools. *Juni Khyat (UGC Care Group I Listed Journal)*, *12*(01).

Rahmat, N. U. R., Riyanto, D. W. I., Alfian, M., & Yuhana, U. M. I. L. (2023). Evaluasi Efisiensi Kinerja Object Relational Mapping pada Web API Point of Sale Menggunakan ISO / IEC 25010. *Jurnal Ilmu Komputer & Agri-Informatika*, *10*.

Rastogi, V. (2015). Software Development Life Cycle Models- Comparison , Consequences. *International Journal of Computer Science and Information Technologies*, *6*(1), 168–172.

Rompis, A. C., & Aji, R. F. (2018). Perbandingan Performa Kinerja Node.js, PHP, dan Python dalam Aplikasi REST. *CogITo Smart Journal*, *4*(1), 171–187. https://doi.org/10.31154/cogito.v4i1.92.171-187

Rosid, M. A. (2017). Implementasi JSON untuk Minimasi Penggunaan Jumlah Kolom Suatu Tabel Pada Database PostgreSQL. *JOINCS (Journal of Informatics, Network, and Computer Science)*, *1*(1), 33. https://doi.org/10.21070/joincs.v1i1.802

Shadiq, J., Safei, A., & Loly, R. W. R. (2021). Pengujian Aplikasi Peminjaman Kendaraan Operasional Kantor Menggunakan BlackBox Testing. *Information Management for Educators and Professionals: Journal of Information Management*, *5*(2), 97. https://doi.org/10.51211/imbi.v5i2.1561

Suzanti, I. O., Fitriani, N., Jauhari, A., & Khozaimi, A. (2020). REST API Implementation on Android Based Monitoring Application. *Journal of Physics: Conference Series*, *1569*(2). https://doi.org/10.1088/1742-6596/1569/2/022088

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.