Jutek : Journal of Usability and Technology Engineering Knowledge

journal homepage: https://ejournal.undar.or.id/index.php/Jutek

Artikel Tinjauan

Menangani Permintaan dan Respons HTTP untuk RESTful API Menggunakan PHP Native

Rizky Parlika [a, *], Arif Nur Cahyo [b]

[a, b] *Program Studi Informatika, Universitas Pembangunan Nasional "Veteran" Jawa Timur, Surabaya, Indonesia*
email: [a] rizkyparlika.if@upnjatim.ac.id

INFO ARTICLE

Kata Kunci:
RESTful API,
Permintaan dan Respons HTTP,
PHP Native,
CRUD,
JSON

ABSTRAK

Pengembangan RESTful API sering kali mengandalkan framework yang menyediakan berbagai fitur otomatis, sehingga mengurangi pemahaman mendasar pengembang terhadap mekanisme HTTP request dan response. Penelitian ini bertujuan untuk mengimplementasikan dan menganalisis penanganan request dan response HTTP secara manual menggunakan PHP native guna memberikan pemahaman yang utuh tentang arsitektur REST dan protokol HTTP. Metode penelitian menggunakan model Waterfall yang terdiri dari analisis kebutuhan, desain sistem, implementasi program, pengujian sistem, dan evaluasi hasil. Hasil penelitian menunjukkan bahwa RESTful API yang dibangun berhasil menangani operasi CRUD (CREATE, READ, UPDATE, DELETE) pada resource "produk" dengan format JSON, menggunakan metode HTTP sesuai standar, serta memberikan respons dengan kode status yang tepat dan penanganan error yang informatif (termasuk status 400, 404, 405, dan 500). Implementasi native memberikan keuntungan berupa kontrol penuh dan pemahaman mendalam terhadap alur komunikasi client-server, meskipun memerlukan lebih banyak kode dan perhatian ekstra pada aspek keamanan serta validasi. Penelitian ini menegaskan bahwa penguasaan konsep dasar HTTP melalui pendekatan native merupakan landasan penting sebelum menggunakan framework, sehingga pengembang dapat membuat keputusan yang lebih tepat dan melakukan debugging yang efektif dalam pengembangan API yang skalabel.

* Corresponding author.
E-mail address: rizkyparlika.if@upnjatim.ac.id (Rizky Parlika).

https://doi.org/xx.xxxx/Jutek.2026.xx.xxx
Received 26 January 2026; Received in revised form 30 January 2026; Accepted 10 February 2026

1. PENDAHULUAN

Perkembangan teknologi informasi yang pesat telah mendorong digitalisasi di berbagai sektor, sehingga sistem berbasis web semakin diandalkan untuk meningkatkan efisiensi dan akurasi proses bisnis (Darsin and Safitri, 2023; Sagala and Yahfizham, 2024). Dalam konteks tersebut, RESTful API menjadi standar utama untuk menghubungkan sistem karena arsitekturnya yang sederhana, fleksibel, dan terstruktur, sehingga mampu memenuhi kebutuhan integrasi data lintas platform (Halomoan, Kharisma, et al., 2021; Simbulan and Aryanto, 2024). Studi terkait pemanfaatan sistem informasi berbasis web pada domain pendidikan, layanan publik, administrasi, hingga komersial menunjukkan bahwa penerapan API secara konsisten meningkatkan kecepatan layanan, akurasi informasi, serta kemampuan sistem untuk mengelola transaksi secara real-time (Esabella, Ramadhan, et al., 2022; Immasari, Mansyur, et al., 2021; Musfikar, Apriadinata, et al., 2023; Novianto and Munir, 2022).

Meskipun Meskipun demikian, banyak pengembang pemula langsung memanfaatkan framework seperti CodeIgniter dan Laravel tanpa memahami mekanisme dasar HTTP, sehingga proses debugging menjadi sulit ketika terjadi kesalahan pada request maupun response (Fallo and Wibowo, 2023). Penelitian terdahulu menekankan pentingnya pemahaman fondasi API sebelum menggunakan framework tingkat lanjut agar pengembang dapat mengelola routing, validasi input, struktur data, dan pemrosesan JSON secara mandiri (Alda, 2023; Nurfadilah, Saputra, et al., 2024; Rusdianto and Billah, 2024; Yusuf and Ahmad, 2026). Berbagai kasus penelitian menunjukkan bahwa sistem web masih menghadapi kendala seperti pencatatan manual, integrasi yang tidak optimal, dan alur data yang belum efisien, sehingga penerapan API menjadi solusi penting dalam membangun layanan digital yang lebih efektif (Harianto, Rifqo, et al., 2024; Kurniawan and Sakti, 2024; Lase and Alasi, 2024; N. A. Putri, Larasati, et al., 2023).

Pemanfaatan RESTful API juga terbukti meningkatkan konsistensi dan skalabilitas sistem pada implementasi aplikasi reservasi, administrasi sekolah, pengelolaan data, inventaris, serta layanan publik berbasis web maupun mobile (Aziz, Sansprayada, et al., 2024; C. Chandra, Wijaya, et al., 2024; Faqihuddin, Taryana, et al., 2025; Maulana, Riady, et al., 2024; Santoso, 2025). Pada berbagai penelitian, pengembangan sistem berbasis API menghasilkan proses komunikasi yang lebih stabil dan mudah dipelihara, terutama ketika backend dibangun secara ringan dan terstruktur menggunakan pendekatan yang fleksibel seperti PHP Native (Aulia, Wahyuni, et al., 2025; S. Chandra and Farisi, 2025; Firdaus and Hidayat, 2025; Hibatullah, Tukino, et al., 2025; Sunenti and Setiawan, 2024). PHP Native memberikan kesempatan bagi pengembang untuk memahami secara menyeluruh bagaimana server menangani request, memetakan endpoint, melakukan sanitasi input, menyusun response, serta memastikan bahwa alur komunikasi berjalan sesuai standar REST (Grahitama, Adiwiguno, et al., 2025; Juarsyah, Ikhwan, et al., 2025; Safana and Kurniawan, 2025).

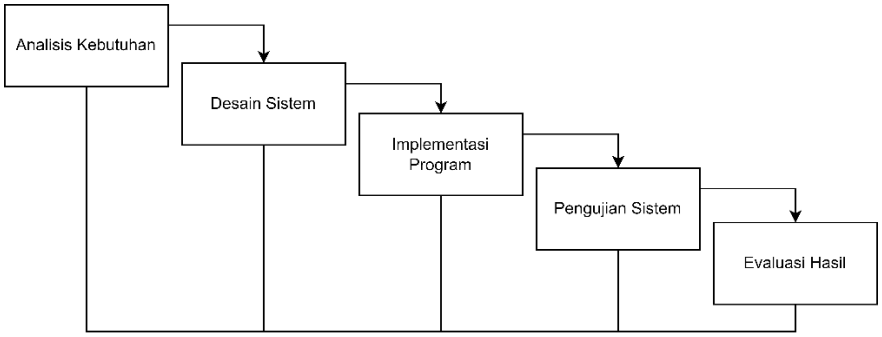
Lebih lanjut, penggunaan metodologi terstruktur seperti Waterfall, Prototype, DevOps, dan SDLC dalam penelitian-penelitian sebelumnya terbukti membantu menghasilkan API yang mudah diuji, memiliki dokumentasi yang jelas, serta memenuhi kebutuhan fungsional dan nonfungsional sistem (Febrian and Ichwani, 2025; Muzaki, Vitianingsih, et al., 2025; K. R. Putri and Mahendra, 2025). Pengujian seperti black-box testing, usability testing, serta verifikasi proses bisnis pada berbagai studi menegaskan bahwa API yang dibangun melalui pendekatan sistematis cenderung lebih stabil, ringkas, dan mudah dikembangkan ulang. Pemahaman mendalam terhadap alur teknis API, mulai dari parsing request, validasi data, hingga penyusunan response JSON menjadi fondasi penting dalam memastikan keberhasilan integrasi sistem pada berbagai implementasi layanan digital.

Berdasarkan kondisi tersebut, penelitian ini disusun untuk memberikan pemahaman komprehensif mengenai proses penanganan HTTP request dan response dalam RESTful API menggunakan PHP Native. Dengan membangun API secara manual, penelitian ini memungkinkan

pengembang memahami alur kerja inti tanpa abstraksi framework, termasuk pembuatan struktur endpoint, konfigurasi routing, teknik validasi input, manajemen error, serta penyusunan response yang sesuai dengan standar REST. Penelitian ini berfokus pada implementasi operasi CRUD untuk resource *produk* sebagai studi kasus, yang mencakup pemrosesan metode GET, POST, PUT, dan DELETE, penyusunan format JSON, serta penerapan HTTP status codes. Dengan batasan tersebut, penelitian ini diharapkan dapat menjadi referensi dasar bagi pengembang dalam memahami konsep RESTful API secara mendalam sebelum beralih ke penggunaan framework modern.

2. METODOLOGI PENILITIAN

Metode penelitian yang digunakan dalam pengembangan RESTful API dengan PHP Native ini mengadopsi model *Waterfall* yang terdiri dari lima tahapan utama: Analisis Kebutuhan, Desain Sistem, Implementasi Program, Pengujian Sistem, dan Evaluasi Hasil. Model ini dipilih karena sifatnya yang sistematis dan terstruktur, memungkinkan setiap fase diselesaikan secara menyeluruh sebelum melanjutkan ke fase berikutnya, seperti yang terlihat pada Gambar 1.



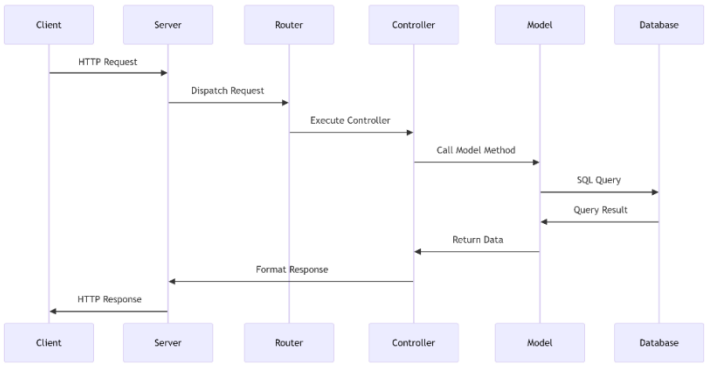
Gambar 1. *Tahapan penelitian metode Waterfall*

2.1 Analisis Kebutuhan

Tahap awal penelitian ini adalah Analisis Kebutuhan, yang bertujuan untuk mengidentifikasi dan mendefinisikan seluruh persyaratan fungsional dan non-fungsional dari sistem RESTful API yang akan dibangun. Proses ini dilakukan melalui studi literatur terkait konsep RESTful API, protokol HTTP, serta praktik implementasi menggunakan PHP Native (Maulana, Riady, et al., 2024; Novianto and Munir, 2022; Simbulan and Aryanto, 2024). Kebutuhan fungsional yang diidentifikasi meliputi kemampuan sistem dalam menangani berbagai metode HTTP (GET, POST, PUT, DELETE), mendukung operasi CRUD pada resource *produk*, serta memberikan respons JSON dengan status code yang sesuai standar REST sebagaimana diterapkan pada berbagai penelitian serupa (Aziz, Sansprayada, et al., 2024; Fallo and Wibowo, 2023; Grahitama, Adiwiguno, et al., 2025). Adapun kebutuhan non-fungsional mencakup performa yang responsif, keamanan data, serta struktur kode yang mudah dipahami dan dipelihara, yang juga menjadi perhatian penting dalam pengembangan sistem informasi berbasis web pada penelitian sebelumnya. Analisis ini menghasilkan spesifikasi teknis yang menjadi dasar bagi tahapan desain dan implementasi API pada penelitian ini.

2.2 Desain Sistem

Berdasarkan hasil analisis kebutuhan, tahap Desain Sistem dilakukan untuk merancang arsitektur dan komponen-komponen perangkat lunak. Desain arsitektur sistem mengikuti pola Layered Architecture yang memisahkan tanggung jawab ke dalam lapisan-lapisan seperti Presentation Layer (Controller), Business Logic Layer (Model), dan Data Access Layer. Diagram alur kerja sistem dirancang untuk menggambarkan proses mulai dari penerimaan HTTP request hingga pengiriman HTTP response, seperti yang terlihat pada Gambar 2.



Gambar 2. *Diagram Sequence Alur Komunikasi RESTful API*

Selain itu, dirancang juga desain database sederhana untuk menyimpan data produk, yang meliputi field seperti *id, name, description, price, stock, created_at,* dan *updated_at* seperti yang terlihat pada Gambar 3.

id	name	description	price	stock	created_at	updated_at
1	Laptop ASUS	Laptop gaming dengan processor Intel i7	12000000.00	15	2025-11-30 07:50:17	2025-12-08 14:23:23
2	Smartphone Samsung	Smartphone dengan kamera 108MP	8000000.00	25	2025-11-30 07:50:17	2025-12-08 14:24:10
3	Headphone Sony	Headphone noise cancelling	2500000.00	30	2025-11-30 07:50:17	2025-12-08 14:21:27
4	Mouse Logitech	Mouse gaming wireless	500000.00	50	2025-11-30 07:50:17	2025-12-08 14:24:28
5	Keyboard Mechanical	Keyboard RGB mechanical switches	1500000.00	20	2025-11-30 07:50:17	2025-12-08 14:24:32

Gambar 3. *Isi Database untuk Resource Produk*

2.3 Implementasi Program

Tahap Implementasi Program dilakukan dengan mengembangkan kode berdasarkan struktur folder yang telah dirancang, seperti yang terlihat pada Gambar 4.

```
restful-api-native/
│
├── index.php
├── config/
│   └── database.php
├── core/
│   ├── Router.php
│   ├── Request.php
│   └── Response.php
├── controllers/
│   └── ProductController.php
├── models/
│   └── Product.php
├── .htaccess
└── README.md
```

Gambar 4. *Struktur Folder Proyek RESTful API PHP Native*

Pada direktori utama restful-api-native, file index.php berperan sebagai pengendali utama yang menerima semua permintaan, sementara folder config, core, controllers, dan models diisi dengan komponen-komponen pendukung yang masing-masing memiliki fungsi spesifik. Seluruh komponen dalam struktur tersebut diintegrasikan untuk membangun API yang utuh. File dalam config menangani koneksi database, folder core berisi pengatur routing serta penanganan request dan response, sedangkan controllers dan models mengelola logika bisnis dan interaksi data. Struktur yang terorganisir ini mempermudah proses pengembangan dan pemeliharaan kode. Kode sumber lengkap dari implementasi ini dapat diakses pada repositori GitHub: https://github.com/aparipip/belajar-restful.git.

2.4 Pengujian Sistem

Setelah implementasi selesai, dilakukan tahap Pengujian Sistem untuk memvalidasi fungsionalitas dan kemampuan API yang dibangun. Pengujian dilakukan secara bertahap dimulai

dari Unit Testing yang menguji setiap fungsi atau metode secara terisolasi seperti fungsi parsing request, pembuatan respons, dan query database. Selanjutnya dilakukan Integration Testing untuk menguji interaksi antar komponen, misalnya alur dari Router ke Controller kemudian ke Model dan Database. Tahap terakhir adalah Functional Testing (Black-Box) yang menguji API dari sisi pengguna akhir dengan mengirimkan permintaan HTTP menggunakan alat Postman. Pengujian ini mencakup semua endpoint (GET, POST, PUT, DELETE) dengan berbagai skenario, termasuk input yang valid, tidak valid, dan kasus error seperti data tidak ditemukan. Hasil dari setiap pengujian dianalisis untuk memastikan sistem berperilaku sesuai dengan spesifikasi yang ditetapkan.

2.5 Evaluasi Hasil

Pada Tahap Evaluasi Hasil ini, kinerja dan capaian keseluruhan sistem dianalisis dengan membandingkan hasil pengujian terhadap tujuan penelitian awal. Aspek yang dievaluasi mencakup kesesuaian fungsional sistem dalam menjalankan semua operasi CRUD sesuai prinsip REST, kualitas respons yang dihasilkan termasuk ketepatan struktur JSON dan kode status HTTP yang informatif, kemampuan sistem dalam menangani kesalahan (*error handling*) dan berbagai skenario input tak terduga, serta pembelajaran yang didapat dari tantangan implementasi manual tanpa framework. Hasil evaluasi ini tidak hanya menjadi kesimpulan penelitian, tetapi juga menjadi bahan rekomendasi untuk pengembangan atau perbaikan lebih lanjut.

3. HASIL DAN PEMBAHASAN

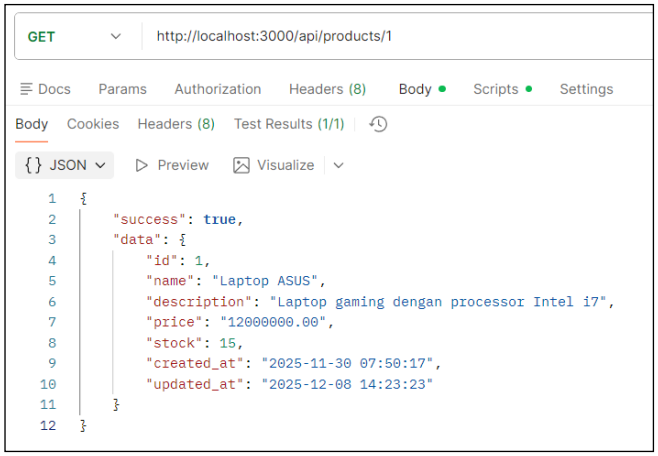
Pada bagian ini memaparkan hasil implementasi dan temuan pengujian dari sistem RESTful API yang dikembangkan menggunakan PHP Native, dilanjutkan dengan analisis mendalam terhadap kinerja sistem, kelebihan, keterbatasan, serta implikasinya dalam konteks pengembangan perangkat lunak berbasis web. Hasil yang disajikan mencakup aspek fungsionalitas, keandalan, dan performa sistem, yang kemudian dibahas secara kritis untuk mengevaluasi pencapaian tujuan penelitian serta membandingkannya dengan studi-studi terdahulu. Pembahasan ini juga mengidentifikasi tantangan teknis yang dihadapi selama pengembangan, sekaligus merumuskan rekomendasi praktis bagi pengembang yang ingin memilih antara pendekatan native dan framework dalam membangun API.

3.1 Hasil Implementasi Sistem

Implementasi RESTful API dengan PHP Native berhasil dibangun dengan struktur yang terorganisir seperti yang ditunjukkan pada Gambar 4. Sistem ini mampu menangani seluruh operasi CRUD untuk resource 'produk' melalui endpoint yang telah ditentukan. Hasil implementasi menunjukkan bahwa API dapat menerima dan memproses HTTP request dengan metode GET, POST, PUT, dan DELETE, serta menghasilkan response JSON yang sesuai dengan standar REST.

3.1.1 Implementasi GET

Gambar 5 menampilkan contoh pengujian endpoint GET /api/products/1 menggunakan Postman yang menunjukkan proses request-response berhasil. Pada sisi kiri gambar terlihat request dengan metode GET yang dikirim ke endpoint http://localhost:3000/api/products/1, sementara pada sisi kanan ditampilkan response JSON dengan status code 200 OK. Struktur response mengikuti format standar REST dengan field success bernilai true yang mengindikasikan operasi berhasil, dilanjutkan dengan objek data yang berisi detail lengkap produk dengan ID 1, termasuk atribut-atribut seperti name, description, price, stock, serta timestamp created_at dan updated_at.

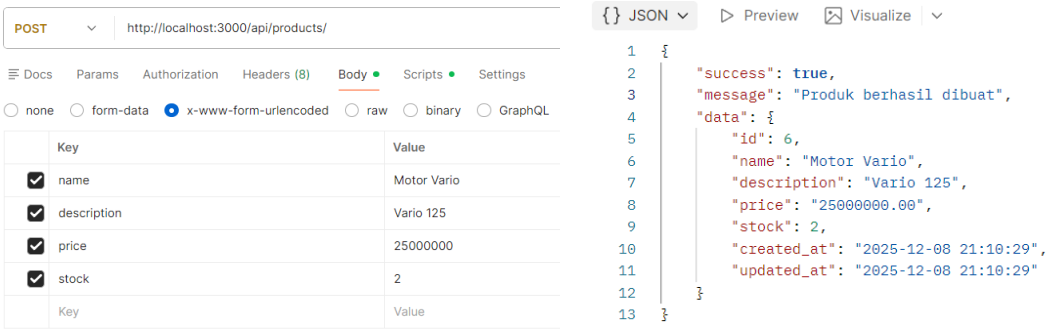


Gambar 5. *Contoh Testing Endpoint GET*

Response yang ditampilkan pada Gambar 5 mengkonfirmasi bahwa implementasi RESTful API berhasil mengambil data spesifik berdasarkan ID dengan struktur JSON yang konsisten dan informatif. Format harga yang ditampilkan dalam bentuk string dengan dua digit desimal ("12000000.00") menunjukkan implementasi format numerik yang tepat untuk data finansial, sedangkan timestamp dalam format YYYY-MM-DD HH:MM:SS memfasilitasi pembacaan dan pemrosesan waktu yang terstandarisasi. Gambar ini secara keseluruhan membuktikan bahwa sistem telah memenuhi prinsip REST dalam menyediakan representasi resource yang lengkap melalui endpoint yang sesuai.

3.1.2 Implementasi POST

Gambar 6 mengilustrasikan proses pengujian operasi CREATE melalui endpoint POST /api/products menggunakan Postman. Bagian atas gambar menampilkan panel request dengan metode POST yang diarahkan ke URL http://localhost:3000/api/products, menggunakan tipe body x-www-form-urlencoded dengan empat parameter: name (Motor Vario), description (Vario 125), price (25000000), dan stock (2). Bagian bawah gambar menampilkan response JSON dengan status code 201 Created yang mengkonfirmasi keberhasilan pembuatan resource baru. Response tersebut memiliki struktur yang konsisten dengan pola REST, mencakup field success bernilai true, message deskriptif, dan objek data yang berisi representasi lengkap produk yang baru dibuat, termasuk id otomatis yang di-generate oleh sistem serta timestamp created_at dan updated_at.



Gambar 6. *Contoh Testing Endpoint POST*

Response pada Gambar 6 menunjukkan bahwa implementasi RESTful API berhasil memproses request pembuatan resource baru sesuai prinsip REST, di mana server mengembalikan status code 201 Created beserta representasi lengkap dari resource yang berhasil dibuat. Struktur response yang informatif dengan pesan sukses dan data lengkap memfasilitasi integrasi dengan client aplikasi. Gambar ini juga mengonfirmasi bahwa sistem telah menerapkan validasi input dasar dan mampu menangani operasi penulisan data ke database dengan baik, sekaligus menunjukkan konsistensi format data numerik (price) dan timestamp yang sesuai dengan standar.

3.1.3 Implementasi PUT

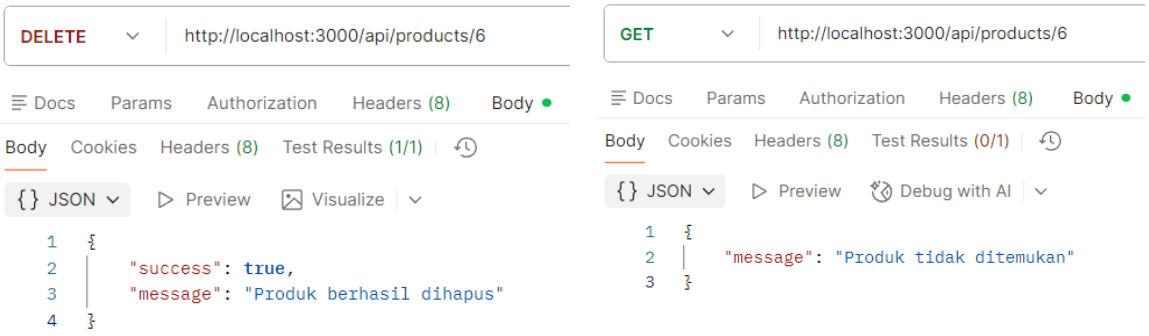
Gambar 7 mendemonstrasikan proses pengujian operasi UPDATE melalui endpoint PUT /api/products/6 menggunakan Postman. Bagian kiri gambar menampilkan panel request dengan metode PUT yang ditargetkan ke URL http://localhost:3000/api/products/6, menggunakan tipe body raw dengan format JSON yang berisi data yang akan diperbarui: name (Motor Vario), description (Vario 125), price (25000000), dan stock (1). Bagian kanan gambar menunjukkan response JSON dengan status code 200 OK yang mengonfirmasi keberhasilan pembaruan resource. Response tersebut memuat field success bernilai true, message yang informatif, serta objek data berisi representasi terbaru produk setelah di-update, termasuk perubahan pada field stock dari 2 menjadi 1 dan pembaruan timestamp pada updated_at yang menyesuaikan dengan waktu proses update.

Gambar 7. *Contoh Testing Endpoint PUT*

Response pada Gambar 7 mengindikasikan bahwa implementasi RESTful API berhasil menerapkan operasi PUT yang bersifat idempoten, di mana multiple identical requests menghasilkan efek yang sama. Perubahan nilai stock dari 2 menjadi 1 dan pembaruan otomatis pada updated_at tanpa mengubah created_at menunjukkan konsistensi dalam mengelola metadata temporal. Gambar ini juga membuktikan bahwa sistem mampu memproses partial update melalui metode PUT dengan format JSON, serta mengembalikan representasi lengkap resource yang telah diperbarui sesuai prinsip uniform interface dalam arsitektur REST.

3.1.4 Implementasi DELETE

Gambar 8 terdiri dari dua bagian yang saling terkait dalam menguji operasi DELETE. Bagian pertama (atas) menampilkan pengujian endpoint DELETE /api/products/6 yang mengirimkan request tanpa body, dengan response JSON berisi success: true dan pesan konfirmasi "Produk berhasil dihapus". Bagian kedua (bawah) menunjukkan verifikasi hasil operasi DELETE melalui request GET /api/products/6 yang mengembalikan response dengan status code 404 Not Found dan pesan "Produk tidak ditemukan". Kedua gambar ini secara bersama-sama mendemonstrasikan siklus lengkap operasi DELETE dalam RESTful API, mulai dari permintaan penghapusan hingga konfirmasi bahwa resource telah benar-benar dihapus dari sistem.



Gambar 8. *Contoh Testing Endpoint DELETE*

Hasil pengujian pada Gambar 8 membuktikan bahwa implementasi RESTful API berhasil menerapkan operasi DELETE sesuai standar REST, di mana resource yang telah dihapus tidak lagi dapat diakses melalui endpoint GET yang sesuai. Response sukses dengan pesan yang informatif pada operasi DELETE memfasilitasi feedback yang jelas bagi client, sementara response 404 Not Found pada verifikasi GET menunjukkan konsistensi state sistem. Gambar ini juga mengkonfirmasi bahwa sistem telah menerapkan prinsip stateless dengan benar, di mana setiap request DELETE bersifat idempoten dan tidak meninggalkan residual data yang dapat diakses setelah operasi berhasil dieksekusi.

3.2 Hasil Pengujian Fungsionalitas

Pengujian fungsionalitas dilakukan terhadap semua endpoint menggunakan Postman. Hasil pengujian menunjukkan bahwa seluruh operasi CRUD berfungsi dengan baik, seperti yang ditampilkan pada Tabel 1.

Tabel 1. *Hasil Pengujian Fungsionalitas Endpoint API*

No	Metode HTTP	Endpoint	Deskripsi	Status Kode Sukses
1	GET	/api/products/{id}	Mendapatkan produk berdasarkan ID	200 OK
2	POST	/api/products	Membuat produk baru	201 Created
3	PUT	/api/products/{id}	Memperbarui produk	200 OK
4	DELETE	/api/products/{id}	Menghapus produk	200 OK

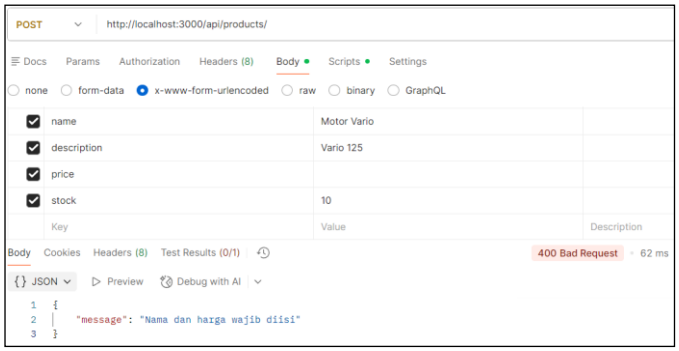
Tabel 1 merangkum hasil pengujian fungsionalitas kelima endpoint utama RESTful API yang diimplementasikan dengan PHP Native. Tabel ini menunjukkan bahwa semua operasi CRUD (Create, Read, Update, Delete) berhasil dijalankan dengan status code HTTP yang sesuai. Implementasi ini mengikuti konvensi RESTful di mana setiap metode HTTP (GET, POST, PUT, DELETE) dipetakan ke operasi spesifik pada resource 'produk', dengan endpoint yang konsisten dan status code yang merefleksikan hasil operasi secara akurat.

3.3 Hasil Pengujian Error Handling

Pengujian Sistem juga dilakukan dengan berbagai skenario error untuk memvalidasi kemampuan penanganan kesalahan. Berikut adalah contoh response error yang dihasilkan:

3.3.1 Response Error 400

Gambar 9 mengilustrasikan skenario pengujian validasi input pada operasi CREATE melalui endpoint POST /api/products. Request yang dikirim menggunakan metode POST dengan body x-www-form-urlencoded memiliki field price yang dikosongkan (nilai kosong), sementara field lainnya seperti name, description, dan stock telah diisi dengan data yang valid. Sistem merespons dengan status code 400 Bad Request dan pesan error JSON yang jelas: "Nama dan harga wajib diisi". Hasil ini menunjukkan bahwa implementasi validasi input berfungsi dengan tepat dalam mendeteksi data yang tidak lengkap sesuai dengan aturan bisnis yang telah ditetapkan.

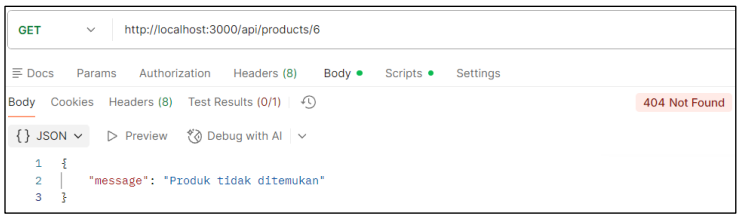


Gambar 9. *Contoh Response Error 400 Bad Request*

Response error pada Gambar 9 membuktikan bahwa sistem telah mengimplementasikan mekanisme validasi input yang sesuai pada level server-side, yang merupakan aspek penting dalam keamanan dan keandalan RESTful API. Pesan error yang deskriptif dan spesifik memfasilitasi debugging dan integrasi yang lebih mudah bagi client aplikasi. Gambar ini juga menunjukkan konsistensi format respons error dengan menggunakan status code HTTP yang tepat (400 untuk client error) dan struktur JSON yang seragam, yang merupakan praktik terbaik dalam pengembangan API yang user-friendly dan mudah dipelihara.

3.3.2 Response Error 404

Gambar 10 menunjukkan skenario pengujian untuk mengakses resource yang tidak tersedia dalam sistem melalui endpoint GET /api/products/6. Request yang dikirim menggunakan metode GET dengan target ID 6 mengembalikan status code 404 Not Found beserta pesan error JSON yang jelas: "Produk tidak ditemukan". Hasil ini membuktikan bahwa sistem telah mengimplementasikan mekanisme validasi keberadaan resource dengan tepat sebelum memproses request lebih lanjut, sesuai dengan prinsip REST yang mengharuskan server memberikan respons yang jujur terkait status resource.

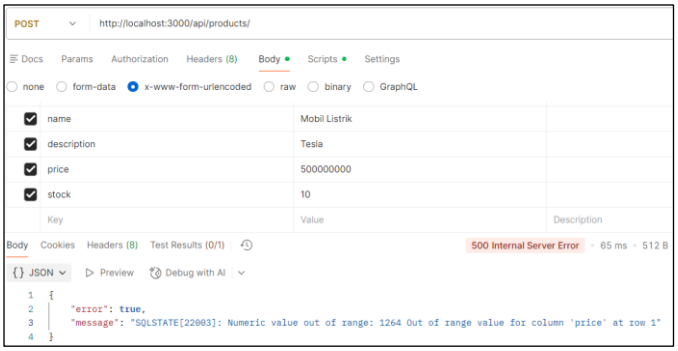


Gambar 10. *Contoh Response Error 404 Not Found*

Response error pada Gambar 10 mengkonfirmasi bahwa sistem RESTful API telah menerapkan penanganan kondisi edge case dengan baik, di mana permintaan terhadap resource yang tidak ada atau telah dihapus mendapatkan respons yang sesuai dengan standar HTTP. Pesan error yang deskriptif dan struktur respons yang konsisten memfasilitasi integrasi yang lebih baik dengan client aplikasi, karena memberikan informasi yang jelas tentang alasan kegagalan request. Gambar ini juga menunjukkan implementasi yang sesuai dari prinsip stateless, di mana setiap request dievaluasi berdasarkan state terkini dari resource tanpa mengandalkan informasi dari request sebelumnya.

3.3.3 Response Error 500

Gambar 11 menunjukkan skenario pengujian yang menghasilkan kesalahan server internal melalui endpoint POST /api/products. Request yang dikirim menggunakan metode POST dengan data yang tampak valid termasuk name (Mobil Listrik), description (Tesla), price (500000000), dan stock (10), ternyata mengakibatkan status code 500 Internal Server Error. Response error JSON memberikan pesan teknis yang detail seperti yang terlihat pada Gambar 11, yang mengindikasikan bahwa nilai yang dimasukkan untuk kolom price melebihi batas yang ditentukan dalam skema database.

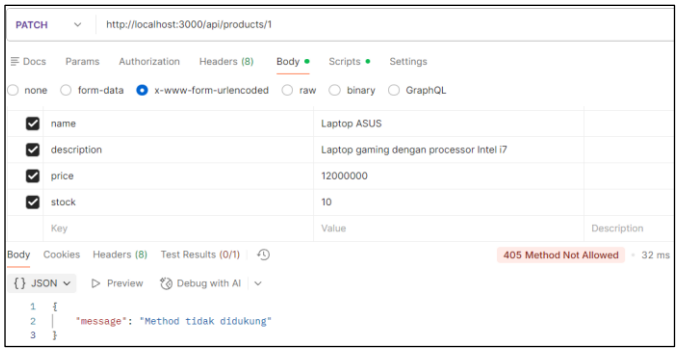


Gambar 11. *Contoh Response Error 500 Internal Server Error*

Response error pada Gambar 11 mengungkapkan kelemahan dalam validasi data pada tingkat aplikasi sebelum permintaan mencapai database. Meskipun sistem telah mengimplementasikan penanganan error yang informatif dengan memberikan pesan teknis yang jelas, error ini seharusnya dapat dicegah dengan validasi yang lebih ketat pada level aplikasi untuk memastikan nilai input sesuai dengan batasan kolom database. Gambar ini juga menunjukkan pentingnya lapisan validasi ganda baik di aplikasi maupun di database untuk menjaga integritas data dan mencegah kegagalan sistem yang tidak terduga.

3.3.4 Response Error 405

Gambar 12 menunjukkan skenario pengujian di mana metode HTTP yang tidak didukung digunakan untuk mengakses endpoint /api/products/1. Request yang dikirim menggunakan metode PATCH dengan data update yang valid (termasuk name, description, price, dan stock) menghasilkan status code 405 Method Not Allowed dan pesan error JSON yang jelas: "Method tidak didukung". Hasil ini membuktikan bahwa sistem secara ketat membatasi operasi hanya pada metode HTTP yang telah didefinisikan dalam implementasi RESTful API, yaitu GET, POST, PUT, dan DELETE.

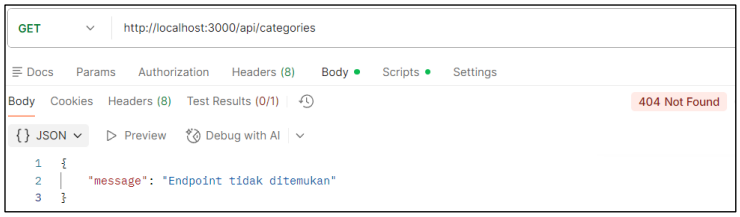


Gambar 12. *Contoh Response Error 405 Method Not Allowed*

Response error pada Gambar 12 mengkonfirmasi bahwa sistem telah mengimplementasikan validasi metode HTTP dengan tepat sesuai dengan spesifikasi RESTful yang hanya mendukung metode standar untuk operasi CRUD. Alih-alih menggunakan PATCH untuk partial update, sistem mengarahkan pengguna untuk menggunakan metode PUT yang telah diimplementasikan untuk operasi update. Gambar ini juga menunjukkan konsistensi dalam penanganan error, di mana metode yang tidak diizinkan mendapatkan respons yang informatif dengan status code HTTP yang tepat (405), memfasilitasi integrasi yang lebih baik dan meningkatkan keamanan sistem dengan membatasi akses hanya pada operasi yang sah.

3.3.5 Response Error 404 (Endpoint Tidak Ditemukan)

Gambar 13 menunjukkan skenario pengujian di mana client mengakses endpoint yang tidak terdaftar dalam sistem melalui GET /api/categories. Request yang dikirim menggunakan metode GET ke URL yang tidak didefinisikan dalam router menghasilkan status code 404 Not Found dan pesan error JSON yang jelas: "Endpoint tidak ditemukan". Hasil ini membuktikan bahwa sistem secara akurat mendeteksi dan menangani permintaan ke rute yang tidak dikenali, mencegah akses ke resource yang tidak sah atau tidak tersedia.



Gambar 13. *Contoh Response Error 404 Not Found (Endpoint Tidak Ditemukan)*

Response error pada Gambar 13 mengkonfirmasi bahwa sistem telah mengimplementasikan mekanisme routing yang sesuai dengan penanganan error yang tepat untuk

endpoint yang tidak terdaftar. Alih-alih mengembalikan halaman default server atau error yang tidak informatif, sistem memberikan respons JSON yang konsisten dengan status code HTTP yang sesuai (404) dan pesan yang jelas. Gambar ini juga menunjukkan konsistensi dalam arsitektur RESTful API, di mana semua error ditangani secara terpusat dan dikembalikan dalam format yang seragam, memudahkan client untuk memahami dan menangani berbagai skenario error dengan cara yang terstandarisasi.

Berdasarkan hasil implementasi dan pengujian yang telah dilakukan, sistem RESTful API yang dibangun menggunakan PHP Native menunjukkan kinerja yang fungsional dan sesuai dengan tujuan pengembangan. Analisis ini menyoroti kelebihan, keterbatasan, serta implikasi praktis dari pendekatan pengembangan tanpa framework dalam konteks aplikasi berbasis web.

Pendekatan PHP Native memberikan fleksibilitas tinggi kepada pengembang karena seluruh proses pengelolaan permintaan dan respons HTTP dilakukan secara manual. Hal ini memungkinkan pemahaman yang lebih mendalam terhadap mekanisme kerja RESTful API serta kebebasan dalam menyesuaikan struktur sistem sesuai kebutuhan spesifik. Selain itu, aplikasi yang dihasilkan bersifat lebih ringan dan efisien karena tidak terbebani fitur framework yang tidak digunakan, serta lebih mudah dideploy tanpa ketergantungan pada pustaka eksternal.

Namun demikian, penggunaan PHP Native juga memiliki keterbatasan. Pengembang harus menulis kode berulang untuk validasi data, penanganan kesalahan, dan pengamanan sistem, yang berpotensi meningkatkan kompleksitas dan risiko kesalahan. Aspek keamanan menjadi perhatian utama karena seluruh mekanisme perlindungan harus dibangun secara manual. Selain itu, pada skala proyek yang lebih besar, pemeliharaan dan pengembangan lanjutan menjadi lebih sulit akibat tidak adanya struktur baku dan fitur siap pakai seperti otentikasi, caching, maupun manajemen basis data.

Dari sisi manfaat, penelitian ini memberikan kontribusi penting terutama dalam bidang pendidikan dan pengembangan perangkat lunak. Hasil penelitian dapat menjadi referensi praktis bagi mahasiswa dan pengembang pemula untuk memahami konsep dasar RESTful API tanpa ketergantungan pada framework. Pendekatan ini juga memudahkan proses analisis dan debugging karena pengembang memiliki pemahaman menyeluruh terhadap alur kerja sistem.

Secara praktis, hasil penelitian menunjukkan bahwa pembangunan RESTful API tanpa framework tetap memungkinkan dan relevan untuk kebutuhan tertentu, khususnya pada sistem berskala kecil hingga menengah atau dengan kebutuhan khusus. Pendekatan ini juga meningkatkan kompetensi teknis pengembang dalam memahami dan mengelola sistem secara mendalam, termasuk saat bekerja dengan framework pada pengembangan selanjutnya.

4 KESIMPULAN

Berdasarkan hasil implementasi dan pengujian yang dilakukan, dapat disimpulkan bahwa penelitian ini sudah berhasil membangun RESTful API fungsional dengan menggunakan PHP native tanpa bantuan framework. API yang dikembangkan mampu menangani seluruh operasi CRUD pada resource “produk” dengan menerapkan metode HTTP yang sesuai (GET, POST, PUT, DELETE), menghasilkan respons dalam format JSON yang terstandar, serta memberikan kode status HTTP yang informatif sesuai dengan prinsip REST. Hasil pengujian fungsionalitas dan penanganan kesalahan menunjukkan bahwa sistem dapat merespons berbagai skenario permintaan dengan baik, termasuk input tidak valid, resource tidak ditemukan, serta kesalahan internal server. Pendekatan native memberikan keuntungan berupa pemahaman mendalam terhadap alur request-response HTTP dan kontrol penuh atas logika aplikasi, meskipun memerlukan penulisan kode yang lebih banyak dibandingkan penggunaan framework. Namun, keterbatasan dalam hal keamanan, validasi yang komprehensif, serta pemeliharaan kode untuk proyek berskala besar menjadi tantangan yang perlu dipertimbangkan. Untuk pengembangan selanjutnya, disarankan untuk menambahkan lapisan keamanan yang lebih mumpuni, mengimplementasikan autentikasi dan otorisasi, serta mengadopsi

komponen middleware untuk meningkatkan modularitas dan keterpeliharaan kode tanpa sepenuhnya bergantung pada framework.

DAFTAR PUSTAKA

Alda. (2023). Pengembangan Aplikasi Pengolahan Data Siswa Berbasis Android Menggunakan Metode Prototyping. *Jurnal Manajemen Informatika (JAMIKA)*, *13*(April), 11–23. https://doi.org/10.34010/jamika.v13i1.8216

Aulia, Wahyuni, et al. (2025). Backend Design and Development of Thesis Management and Scheduling Application Using Rule-Based Algorithm in Physics Department UPN "Veteran" Jatim. *JURNAL TEKNOLOGI DAN OPEN SOURCE*, *8*(1), 153–163. https://doi.org/10.36378/jtos.v8i1.4386

Aziz, Sansprayada, et al. (2024). Perancangan Sistem Adminisatrasi Penjualan pada PT SurMoRin dengan Menggunakan PHP dan MYSQL. *Jurnal Minfo Polgan*, *13*, 1641–1650. https://doi.org/https://doi.org/10.33395/jmp.v13i2.14148 e-ISSN

Chandra, C., Wijaya, et al. (2024). Perancangan dan Implementasi RESTful API untuk Aplikasi Mobile Pembelajaran Flora dan Fauna pada Google Cloud Platform. *Jurnal Sains Teknologi Dan Sistem Informasi*, *4*(1), 58–69. https://doi.org/10.54259/satesi.v4i1.2850

Chandra, S., & Farisi. (2025). Comparative Analysis of RESTful , GraphQL , and gRPC APIs : Perfomance Insight from Load and Stress Testing. *Jurnal SISFOKOM*, *14*, 81–85. https://doi.org/10.32736/sisfokom.v14i1.2315

Darsin, & Safitri. (2023). RANCANG BANGUN SISTEM INFORMASI HOTEL SARBINI MENGGALA TULANG BAWANG MENGGUNAKAN PHP MYSQL. *Jurnal Teknologi Dan Informatika (JEDA)*, *4*(1), 1–8. https://doi.org/https://doi.org/10.57084/jeda.v4i1.1159

Esabella, Ramadhan, et al. (2022). Sistem Informasi Pelayanan Jasa Notaris Menggunakan PHP dan MySQL di Kantor Notaris Kusliana S.H., M.Kn. *Bulletin of Information Technology (BIT)*, *3*(1), 6–9. https://doi.org/10.47065/bit.v3i1.173

Fallo, & Wibowo. (2023). Penerapan REST API Untuk Aplikasi Reservasi Dokter Praktik Berbasis Android (Studi Kasus: Klinik dr. Candra Safitri). *TEKNIKA*, *12*(2), 106–114. https://doi.org/10.34148/teknika.v12i2.615

Faqihuddin, Taryana, et al. (2025). PENGEMBANGAN APPLICATION PROGRAMMING INTERFACE (API) APLIKASI PENDETEKSI PENYAKIT TANAMAN MENGGUNAKAN METODE DEVOPS. *Jurnal Sistem Informasi Dan Teknologi Komputasi*, *2*(April), 66–75. https://doi.org/https://doi.org/10.61124/sinta.v2i2.42

Febrian, & Ichwani. (2025). Rancang Bangun Aplikasi Booking Online Layanan Potongan Rambut Berbasis Website Menggunakan REST API. *IKRAITH-INFORMATIKA*, *9*(3), 63–70. https://doi.org/https://doi.org/10.37817/ikraith-informatika.v9i3

Firdaus, & Hidayat. (2025). Perancangan dan Implementasi Sistem Absensi Siswa Berbasis Web Menggunakan Face Recognition dan SMS Gateway. *Jurnal Manajemen Informatika (JAMIKA)*, *15*(April), 32–46. https://doi.org/10.34010/jamika.v15i1.13601

Grahitama, Adiwiguno, et al. (2025). Design and Implementation of a RESTful API-Based Point of Sale System. *NUANSA INFORMATIKA*, *19*. https://doi.org/https://doi.org/10.25134/ilkom.v19i1.343

Halomoan, Kharisma, et al. (2021). Pengembangan Domain Specific Language Untuk Aplikasi CRUD Berbasis Web. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *5*(1), 34–41.

Harianto, Rifqo, et al. (2024). Application Of Programming Logic And Algorithm To Improve Data Security Of Rantau Panjang Village Citizens In The Village Information System. *SINTA JOURNAL*, *5*(2), 145–152. https://doi.org/https://doi.org/10.37638/sinta.5.2.145-152

Hibatullah, Tukino, et al. (2025). PERANCANGAN WEBSITE E-COMMERCE MENGGUNAKAN METODE WATERFALL PADA PENJUALAN ALAT KOMPUTER. *Jurnal Sistem Informasi Dan Teknologi Komputasi*, *2*, 116–124. https://doi.org/https://doi.org/10.61124/sinta.v2i3.61

Immasari, Mansyur, et al. (2021). RANCANG BANGUN SISTEM INFORMASI PENGOLAHAN DATA NILAI SISWA SMP ABC MENGGUNAKAN PHP dan MySQL. *Journal of Information System, Applied, Management, Accounting and Research.*, *5*(1), 236–248. https://doi.org/https://doi.org/10.52362/jisamar.v5i1.372

Juarsyah, Ikhwan, et al. (2025). Implementasi Metode Prototyping pada Aplikasi Pembelajaran Metode Tikrar Dalam Menghafal Al-Quran Berbasis Android. *Jurnal Manajemen Informatika (JAMIKA)*, *15*(April), 1–16. https://doi.org/10.34010/jamika.v15i1.13423

Kurniawan, & Sakti. (2024). IMPLEMENTASI WEB SERVICE MENGGUNAKAN RESTFUL API UNTUK INTEGRASI DATA MINECRAFT SERVER PADA APLIKASI REFORGED WORLD. *JMIK (JURNAL MAHASISWA ILMU KOMPUTER)*, *5*(2). https://doi.org/https://doi.org/10.24127/ilmukomputer.v5i2.7286

Lase, & Alasi. (2024). Penerapan Web untuk Pengolahan Data Pegawai Kantor Desa Menggunakan Bahasa Pemrograman PHP dan UML. *JURNAL MAHAJANA INFORMASI*, *9*(1), 1–6. https://doi.org/https://doi.org/10.51544/jurnalmi.v9i1.5052

Maulana, Riady, et al. (2024). Perancangan Backend Api Berbasis Rest-Api pada Aplikasi Rekomendasi Resep Makanan (Rest-Api. *Mars : Jurnal Teknik Mesin, Industri, Elektro Dan Ilmu Komputer*, *2*(3). https://doi.org/https://doi.org/10.61132/mars.v2i3.137

Musfikar, Apriadinata, et al. (2023). Aplikasi Prediksi Prestasi pada Siswa Menggunakan Algoritma C4.5. *Jurnal Manajemen Informatika (JAMIKA)*, *13*, 148–162. https://doi.org/10.34010/jamika.v13i2.10649

Muzaki, Vitianingsih, et al. (2025). SISTEM INFORMASI PERSEDIAAN STOK BERDASARKAN TURNOVER RATIO. *Journal of Information System Management (JOISM*, *7*(1). https://doi.org/https://doi.org/10.24076/joism.2025v7i1.2120

Novianto, & Munir. (2022). ANALISIS DAN IMPLEMENTASI RESTFUL API GUNA PENGEMBANGAN SISTEM INFORMASI AKADEMIK PADA PERGURUAN TINGGI. *Jurnal Informatika Terpadu*, *8*(1), 47–61. https://doi.org/https://doi.org/10.54914/jit.v8i1.409

Nurfadilah, Saputra, et al. (2024). RANCANG BANGUN APLIKASI KONVERSI FILE EXCEL KE FILE KML BERBASIS WEB. *Jurnal Teknologi Informasi, Komputer Dan Aplikasinya (JTIKA)*, *6*(1), 316–323. https://doi.org/https://doi.org/10.29303/jtika.v6i1.353

Putri, K. R., & Mahendra. (2025). Rancang Bangun Sistem Informasi Sewa Sepeda Berbasis Web di Kota Singaraja. *Journal of Artificial Intelligence and Digital Business (RIGGS)*, *4*(3), 7187–7197. https://doi.org/https://doi.org/10.31004/riggs.v4i3.3066

Putri, N. A., Larasati, et al. (2023). Sistem Informasi Inventaris Barang Berbasis Web menggunakan Codeigniter pada Pusat Pendidikan dan Pelatihan Pajak (PPPP). *Jurnal Sistem Komputer Dan Kecerdasan Buatan*. https://doi.org/https://doi.org/10.47970/siskom-kb.v7i1.475

Rusdianto, & Billah. (2024). RANCANG BANGUN SISTEM INFORMASI KAS BERBASIS WEB UNTUK MENINGKATKAN AKURASI DATA MENGGUNAKAN FRAMEWORK CODEIGNITER (STUDI KASUS: RUMAH MAKAN AMPERA DADAKAN). *Jurnal Sistem Informasi (J-SIKA)*, *06*, 55–66.

Safana, & Kurniawan. (2025). IMPLEMENTASI PEMROGRAMAN WEB UNTUK MENINGKATKAN EFISIENSI LAYANAN INFORMASI Alivia. *JURNAL RISET TEKNIK KOMPUTER*, *2*(3), 117–123. https://doi.org/https://doi.org/10.69714/k7bh3m03

Sagala, & Yahfizham. (2024). Analisis Pengenalan Konsep Algoritma Pemrograman Matematika Pada Kehidupan Sehari Hari. *Jurnal Ilmu Pendidikan, Bahasa, Sastra Dan Budaya (MORFOLOGI)*, *2*(1). https://doi.org/https://doi.org/10.61132/morfologi.v2i1.267

Santoso. (2025). Implementasi Teknologi Frontend Modern pada Website Yellowweb: Kolaborasi Booostrap 5 Framework dan jQuery. *JUMIN*, *6*(2), 873–883. https://doi.org/https://doi.org/10.55338/jumin.v6i2.5043

Simbulan, & Aryanto. (2024). Implementasi REST API Web Services pada Aplikasi Sumber Daya Manusia. *Jurnal Indonesia : Manajemen Informatika Dan Komunikasi*, *5*(1), 552–560. https://doi.org/https://doi.org/10.35870/jimik.v5i1.511

Sunenti, & Setiawan. (2024). Sistem Informasi Inventaris Berbasis Web Pada Kelurahan Rancabolang. *Jurnal Teknologi Sistem Informasi*, *5*(July), 59–71. https://doi.org/10.35957/jtsi.v5i2.8554

Yusuf, & Ahmad. (2026). Sales Information System Utilizing 13.56 MHz RFID Member Cards for Enhanced Efficiency in Cooperative Stores. *Penelitian Ilmu Komputer, Sistem Embedded and Logic*, *12*(225), 149–160. https://doi.org/10.33558/piksel.v12i1.9334