Perancangan Backend Api Berbasis Rest-Api pada Aplikasi Rekomendasi Resep Makanan

by Agil Maulana

Submission date: 14-Jun-2024 10:58PM (UTC-0500)
Submission ID: 2402811465
File name: MARS_-_VOLUME_2,_NO._3_JUNI_2024_hal_94-106.docx (1.5M)
Word count: 2505
Character count: 16496

Mars : Jurnal Teknik Mesin, Industri, Elektro Dan Ilmu Komputer
Vol.2, No.3 Juni 2024
e-ISSN: 3031-8742; p-ISSN: 3031-8750, Hal 94-106
DOI: https://doi.org/10.61132/mars.v2i3.

Perancangan *Backend* Api Berbasis *Rest*-Api pada Aplikasi Rekomendasi Resep Makanan

(*Rest-Api Based Backend Api Design In Food Recipe Recommendation Application*)

Agil Maulana Nan Riady[1], Paniran Paniran [2], I Made Budi Suksmadana[3]
[1,2&3] Jurusan Teknik Elektro Universitas Mataram
Jl. Majapahit 62, Mataram, Lombok Nusa Tenggara Barat
Email: ragilthejester@gmail.com[1], paniranmt@yahoo.com[2], mdbudisuk@unram.ac.id[3]

Abstract. *This study explores the design intricacies of a scalable RESTful Backend API tailored for a sophisticated food recipe recommendation application, with a primary focus on leveraging the Google Cloud Platform (GCP) for deployment. Employing a service-oriented paradigm, the API segregates Backend functionalities, fostering optimal scalability. The REST architecture ensures seamless integration, while GCP guarantees reliability and scalability. Utilizing Node.js and Express.js, the API efficiently manages culinary recipes and user preferences. Rigorous performance evaluations affirm its rapid responsiveness. This paper offers pragmatic guidelines for developers, emphasizing the significance of GCP for seamless and scalable deployments.*

Keyword **:** *Back-end*, API, REST-API, *Google Cloud Platform*

Abstrak. Studi ini mengeksplorasi seluk-beluk desain RESTful Backend API yang dapat diskalakan dan disesuaikan untuk aplikasi rekomendasi resep makanan canggih, dengan fokus utama memanfaatkan Google Cloud Platform (GCP) untuk penerapannya. Dengan menggunakan paradigma berorientasi layanan, API memisahkan fungsi Backend, sehingga mendorong skalabilitas optimal. Arsitektur REST memastikan integrasi yang lancar, sementara GCP menjamin keandalan dan skalabilitas. Memanfaatkan Node.js dan Express.js, API secara efisien mengelola resep kuliner dan preferensi pengguna. Evaluasi kinerja yang ketat menegaskan kecepatan tanggapnya. Makalah ini menawarkan pedoman pragmatis bagi developer, yang menekankan pentingnya GCP untuk penerapan yang lancar dan terukur.

Kata Kunci : Back-end, API, REST-API, Google Cloud Platform

PENDAHULUAN

Backend adalah sisi *server* aplikasi yang tidak terlihat oleh pengguna. *Backend* bertanggung jawab untuk logika aplikasi, data, dan komunikasi dengan *database*. *Backend* memiliki beberapa fungsi utama, yaitu menyimpan dan mengelola data, memproses permintaan, menyediakan layanan, dan menjaga keamanan. *Backend* berbeda dengan frontend dalam beberapa hal, yaitu visibilitas, tanggung jawab, dan bahasa pemrograman yang digunakan[1], sehingga *back-end* tidak melakukan interaksi secara langsung kepada pengguna. Dalam interaksi antar layanan *client* yang berbeda, salah satu teknologi yang digunakan adalah *Application Programming Interfaces* (API). API (*Application Programming Interface*) adalah antarmuka yang memungkinkan aplikasi untuk berkomunikasi dan berinteraksi satu sama lain. API menyediakan standar dan spesifikasi untuk pertukaran data, fungsi, dan layanan[2]. API

Received: Mei 30, 2024; Accepted: Juni 15, 2024; Published: Juni 30, 2024
* *Paniran , paniranmt@yahoo.com*

memfasilitasi pengembang untuk menggunakan fitur yang telah ada di aplikasi lain, sehingga menghemat waktu dan tenaga.

Salah satu arsitektur *back-end* adalah *Representational State Transfer* (REST). REST adalah kumpulan aturan desain yang mengatur bagaimana data ditransfer melalui antarmuka standar seperti HTTP. REST API adalah istilah untuk layanan web yang menggunakan arsitektur REST sebagai antarmuka pemrograman aplikasi (API)[3].

Untuk membangun sebuah *back-end server*, diperlukan penggunaan bahasa pemrograman yang berjalan pada sisi *server* (*server-side*). Sebelumnya, Bratakusuma, Azmi, dan Ayuningtiyas meneliti pembangunan sistem *back-end* menggunakan API REST untuk Aplikasi Alat Tulis Kantor Bank Indonesia di Purwokerto. Mereka memilih Node.js sebagai *platform* pengembangan karena portabel di berbagai sistem operasi tanpa perlu mengubah kode program[4]. Hasanuddin, Asgar, dan Hartono juga meneliti desain *back-end server* menggunakan arsitektur REST dan Node.js untuk REST API Aplikasi Weshare, *platform* donasi kemanusiaan. Alasan memilih Node.js mirip dengan penelitian sebelumnya, yaitu portabilitas dan kemudahan penggunaan karena Node.js memiliki pustaka *server* HTTP internal yang tidak memerlukan program *server* web seperti Apache atau Nginx. [5].

Beralih dari hal tersebut, Node.js adalah *platform* pengembangan *open-source* yang memungkinkan *programmer* untuk membangun aplikasi *server*-side dengan menggunakan bahasa pemrograman JavaScript. Node.js didasarkan pada *runtime* JavaScript V8 Chrome yang dirancang untuk performa dan skalabilitas. [6]. Keunggulan Node.js terletak pada teknik *non-blocking*. Teknik ini memungkinkan banyak operasi dijalankan secara bersamaan tanpa harus menunggu operasi sebelumnya selesai. Hal ini memungkinkan Node.js untuk menangani banyak permintaan secara paralel dengan efisien. [7].

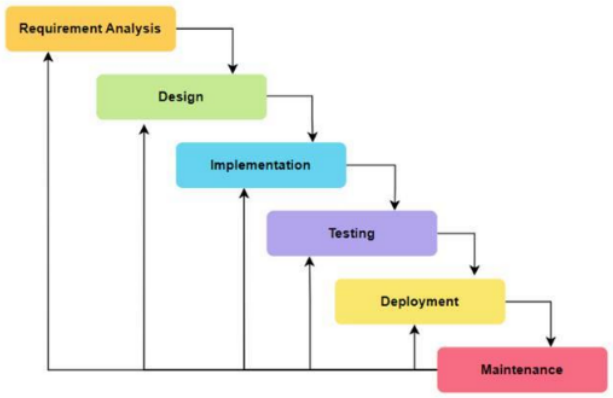
aplikasi rekomendasi resep makanan "PindaiRasa" merupakan aplikasi *mobile* inovatif yang memiliki tujuan utama untuk menyediakan resep makanan berdasarkan stok makanan yang tersedia di dapur pengguna. aplikasi ini diharapkan mengurangi pemborosan bahan makanan, meningkatkan efisiensi pengguna dalam merencanakan menu sehari-hari, serta memberikan inspirasi baru untuk menciptakan hidangan lezat dan bergizi.

Aplikasi PindaiRasa memerlukan sistem *back-end* yang kuat untuk mendukung operasinya. Sistem ini akan menangani logika bisnis, penyimpanan dan manipulasi data, serta komunikasi antara aplikasi *mobile*, *model machine learning*, dan *server*. *Back-end* yang andal dan efisien akan memastikan pengalaman pengguna yang optimal dan membantu mereka memilih menu makanan yang tepat. Oleh karena itu, penelitian ini berfokus pada perancangan sistem *back-end* dan arsitektur REST API untuk PindaiRasa, menggunakan *framework* Node.js

Express. Tujuan utama adalah membangun *back-end* yang efektif dan fungsional untuk aplikasi *mobile* ini..

METODE PENELITIAN

Penelitian ini menggunakan metode *Waterfall* untuk membangun dan merancang sistem *back-end*. Metode ini bersifat berurutan dan bertahap, di mana setiap langkah harus diselesaikan secara menyeluruh sebelum beralih ke langkah berikutnya [8]. Gambar 1 menggambarkan tahapan-tahapan *Waterfall* yang diterapkan dalam penelitian ini, yaitu:



Gambar 1. Tahapan Metode *Waterfall* pada Perancangan *Back-end* Aplikasi *Mobile* PindaiRasa

1. Analisa kebutuhan(*requirement analysis*): langkah awal dalam proses ini adalah mengumpulkan semua informasi yang relevan, baik berupa dokumen maupun antarmuka, untuk memahami kebutuhan pengguna. Informasi ini kemudian dianalisis untuk menentukan solusi perangkat lunak yang sesuai dengan kebutuhan tersebut.
2. Desain (*Design*): berdasarkan analisis kebutuhan di tahap awal, tahap desain berfokus pada perancangan arsitektur sistem yang terstruktur. Hasil dari tahap ini adalah berbagai diagram yang menggambarkan sistem, seperti *use case diagram*, *sequence diagram*, dan *class diagram*, serta rancangan struktur *database* untuk aplikasi *mobile* PindaiRasa.
3. Implementasi (*Implementation*): berdasarkan desain arsitektur sistem yang telah disusun, langkah berikutnya adalah menerjemahkannya menjadi kode pemrograman yang konkret. Implementasi ini difokuskan pada pengembangan bagian *back-end* aplikasi dengan menggunakan *framework* Node.js Express. Untuk pengembangan ini, penulis memanfaatkan *Google Cloud Platform* (GCP) sebagai arsitektur

pengembangan. GCP adalah layanan *cloud computing* publik dari Google yang menyediakan berbagai layanan *hosting*, termasuk komputasi, penyimpanan, dan pengembangan aplikasi yang berjalan di atas infrastruktur perangkat keras Google [9].

4. Pengujian (*Testing*): setelah implementasi kode selesai, langkah berikutnya adalah pengujian sistem. Pada tahap ini, penulis menggunakan metode *black box testing*, yaitu metode pengujian yang berfokus pada fungsionalitas perangkat lunak [10]. Metode ini dipilih untuk memastikan semua *endpoint* yang telah dibuat berfungsi dengan baik.
5. Deploy (*Deployment*): pada tahap ini, penulis melakukan *deployment* agar API *back-end* yang telah dikembangkan dapat digunakan oleh tim pengembangan aplikasi *mobile*.
6. Pemeliharaan (*Maintenance*): tahapan terakhir adalah pemeliharaan *back-end* yang telah dikembangkan untuk menemukan *error/bug* dalam aplikasi dan memastikan keseluruhan sistem berjalan secara optimal.

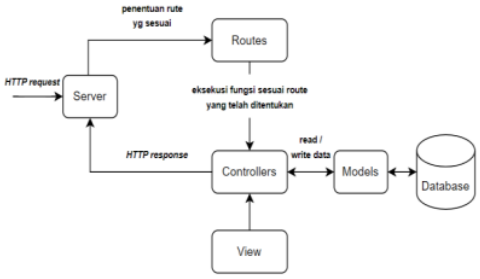
HASIL DAN PEMBAHASAN

Analisa kebutuhan (*requirement analysis*)

Setelah melakukan analisis kebutuhan, ditemukan beberapa fitur yang ingin diimplementasikan dalam aplikasi *mobile* PindaiRasa. Fitur-fitur tersebut mencakup registrasi dan *login* pengguna untuk mengelola akses ke aplikasi. Selain itu, ada juga fitur favorit pengguna yang memungkinkan mereka menambahkan resep makanan favorit, serta fitur *editUser* yang memungkinkan pengguna mengatur data profil mereka. Selanjutnya, terdapat fitur Scan bahan makanan yang memungkinkan pengguna memotret bahan makanan menggunakan kamera ponsel, kemudian akan diproses oleh model *machine learning* yang dikembangkan oleh tim pengembang *machine learning*. Model tersebut akan mencari resep makanan yang sesuai dengan bahan yang telah difoto. Terakhir, ada fitur *chatbot* yang memungkinkan pengguna berkomunikasi dengan aplikasi untuk berkonsultasi terkait resep makanan dan kandungan gizi yang terdapat dalam makanan tersebut.

1. Desain (*Design*)

   a. Rancangan arsitektur sistem



Gambar 2. Rancangan Arsitektur *Back-end*

Dalam perancangan back-end aplikasi mobile PindaiRasa, terdapat beberapa komponen penting, yaitu database, models, controllers, routes, dan server. Komponen-komponen ini bekerja sama untuk memastikan aplikasi berfungsi secara optimal. Pada tahap awal, server berfungsi sebagai penghubung antara client dan aplikasi, menerima request dari client, dan menentukan rute (routes) berdasarkan metode permintaan yang diterima.

Selanjutnya, controllers mengeksekusi fungsi-fungsi yang sesuai dengan rute tersebut. Controllers berinteraksi langsung dengan database melalui models untuk mengambil dan memanipulasi data yang diperlukan [11]. Hasilnya kemudian dikirimkan kembali ke client sebagai response yang sesuai dengan permintaan. Desain arsitektur sistem back-end aplikasi PindaiRasa dapat dilihat pada Gambar 2.

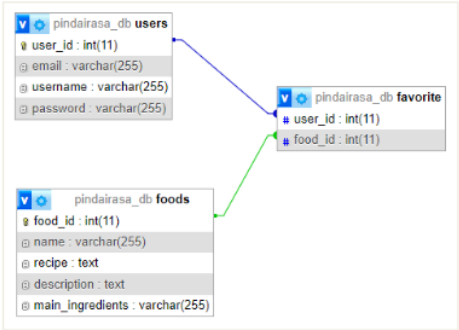
   b. *Use Case Diagram*

Gambar 3. *Use Case Diagram*

c. *Class Diagram*

Gambar 4. *class diagram*

d. Struktur *Database*



Gambar 5. Struktur *Database*

Pada Gambar di atas, struktur *database* dari rancangan *back-end* aplikasi *mobile* PindaiRasa terdiri dari tiga tabel yaitu tabel users, tabel favorite, dan tabel foods. Id_user dan id_food menjadi *foreign key* pada tabel favorite. Ketiga tabel ini memiliki relasi yang saling bergantung satu sama lain. Relasi antara tabel users dan favorite adalah *1 to 1*, di mana satu pengguna hanya dapat membuat satu daftar resep favorit dan satu daftar favorit hanya dapat dimiliki oleh satu pengguna. Selanjutnya, relasi antara tabel *favorite* dan *foods* adalah *many to many*, di mana satu daftar favorit dapat memiliki lebih dari satu resep makanan, dan sebaliknya satu resep makanan dapat masuk ke dalam lebih dari satu daftar favorit.

Implementasi (*Implementation*)

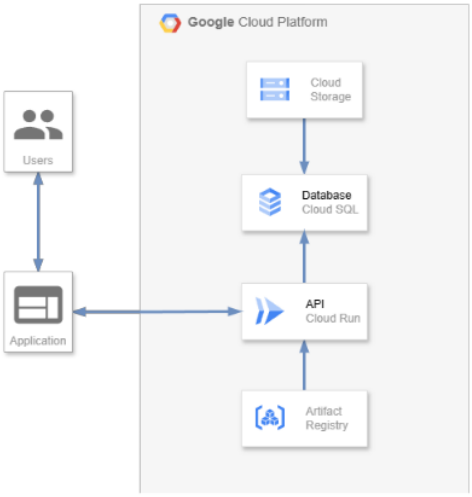
Langkah pertama yang dilakukan oleh penulis adalah merancang kebutuhan *endpoint* yang diperlukan. Proses ini melibatkan komunikasi dengan tim pengembang *mobile* untuk memahami kebutuhan API yang harus dibuat berdasarkan fitur-fitur yang ada. Setelah diskusi, beberapa API yang perlu dikembangkan telah diidentifikasi sebagai berikut:

e-ISSN: 3031-8742; p-ISSN: 3031-8750, Hal 94-106

Tabel 1 API untuk aplikasi pindaiRasa

Users	favorite	foods
registerUser()	getFavorite()	getFoods()
loginUser()	addFavorite()	
editUser()	deleteFavorite()	

Penulis kemudian mempersiapkan kebutuhan *hardware* dan *software* yang diperlukan, serta memasang Node.js Express dan beberapa paket tambahan seperti *body-parser, cookie-parser, sequelize*, dan lainnya. Untuk membuat endpoint proses login pengguna, penulis menggunakan JSON Web Token (JWT) sebagai metode autentikasi. Mekanisme JWT mirip dengan kata sandi; setelah login berhasil, *server* menghasilkan token yang kemudian disimpan di penyimpanan lokal atau *cookie browser*. [12].



Gambar 6. Skema Arsitektur Aplikasi Pada *Google Cloud Platform*

Terdapat beberapa layanan GCP yang digunakan dalam proses implementasi *back-end* ini. Pertama, terdapat *Cloud Run Service* yang digunakan untuk melakukan *deployment model Machine Learning* dan *Back-end* API. Model dan API dibungkus ke dalam *container* menggunakan *docker* dan disimpan ke dalam *Artifact Registry*[13]. Cloud SQL (MySQL) digunakan sebagai database relasional yang dapat diakses secara *realtime*, sementara *Cloud Storage* digunakan untuk menyimpan data statis seperti foto makanan. Skema arsitektur layanan GCP dapat dilihat pada Gambar 6

Pengujian (*Testing*)

Hasil pengujian dengan menggunakan *black box testing* :

Tabel 2 hasil testing untuk table users

No	API	Kondisi	Hasil	Status
1	registerUser	Seluruh data terisi	Registrasi berhasil	Sukses
2		Sebagian data tidak diisi	Registrasi gagal	Sukses
3		Email yang digunakan belum terdaftar	Registrasi berhasil	Sukses
4		Email yang digunakan sudah terdaftar	Pengguna diminta menggunakan email lain	Sukses
5	loginUser	Email & Password benar	Login berhasil dan diarahkan ke beranda	Sukses
6		Email / Password Salah	Login gagal	Sukses
7		email / password tidak diisi	Login gagal dan muncul pop up silahkan isi data dengan lengkap	
8		Email belum terdaftar	Pengguna diarahkan untuk registrasi akun	Sukses
9	editUser	Data pengguna diubah	Data berhasil diubah	Sukses
10		Pengguna memasukan email yang telah terdaftar untuk pengguna lain	Muncul pop up pesan email telah telah terdaftar	Sukses

Tabel 3 hasil testing untuk table favorite

No	API	Kondisi	Hasil	Status
1	getFavorite	Resep makanan ditemukan	Resep favorit ditampilkan	Sukses
2		Resep makanan tidak ditemukan	Muncul pop up resep tidak ditemukan	Sukses
3	addFavorite	Proses menambahkan resep favorit berhasil	List favorit dibuat dan muncul pop up resep berhasil ditambahkan	Sukses
4		Proses menambahkan resep favorit gagal	Muncul pop up gagal menambahkan resep favorit	Sukses
5	DeleteFavorite	Proses menghapus resep favorit berhasil	Resep terkait dihapus dari list dan muncul pop up resep berhasil dihapus	Sukses
6		Proses menghapus resep favorit gagal	Resep terkait tetap berada dalam list dan muncul pop up gagal menghapus resep	sukses

Tabel 4 hasil testing untuk table foods

No	API	Kondisi	Hasil	Status
1	getFoods	Resep makanan ditemukan	Resep makanan ditampilkan	Sukses
2		Resep makanan tidak ditemukan	Muncul pop up resep tidak ditemukan	Sukses

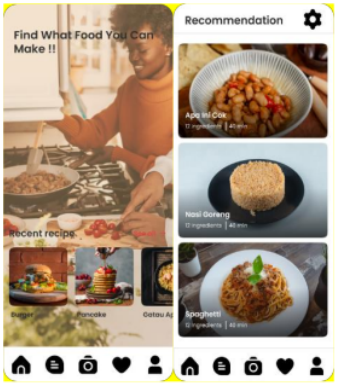
Deploy (*Deployment*)

Penulis menggunakan layanan *Cloud Run* di GCP sebagai media *deployment* untuk menyebarkan API, sehingga tim *mobile development* dapat mengaksesnya dengan mudah. *Cloud Run* adalah layanan komputasi tanpa server yang memungkinkan penulis untuk menjalankan API tanpa harus mengelola infrastruktur. Hasilnya bisa dilihat di Gambar 7, yang menunjukkan API PindaiRasa yang telah berhasil diluncurkan dan menampilkan hasil consume API oleh tim *mobile development* dimana proses pengambilan dan manipulasi data dari *database* telah berhasil dilakukan.

Gambar 7. *Deploy* Pada *Cloud Run*

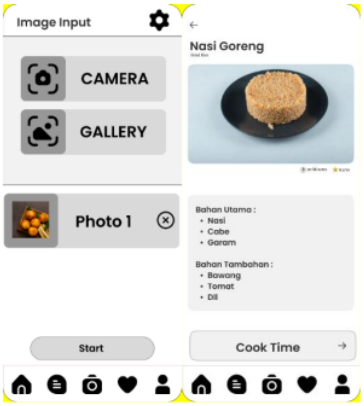
Gambar 8 sampai 11 berikut merupakan hasil tampilan front-end dari aplikasi PindaiRasa yang berhasil terhubung dengan back-end yang di deploy ke GCP.

Gambar 8. halaman login dan register

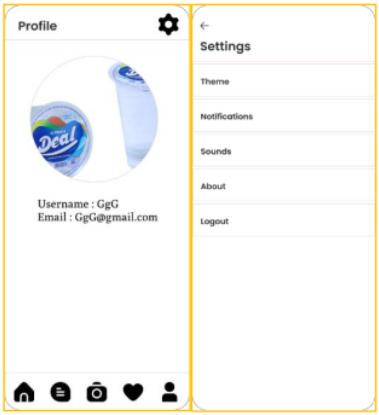


Gambar 9. halaman beranda

e-ISSN: 3031-8742; p-ISSN: 3031-8750, Hal 94-106



Gambar 10. halaman scan bahan makanan



Gambar 11. halaman akun dan pengaturan

Pemiliharaan (*maintenace*)

Proses maintenance dimulai dengan melakukan beberapa langkah maintenance secara berkala, seperti memantau kredit GCP dan kuota penyimpanan database Cloud SQL. Hal ini penting untuk memastikan ketersediaan kredit yang cukup untuk penggunaan berbagai layanan GCP dan menghindari terhentinya layanan karena memori database yang penuh. Selain itu, dengan memantau memori database, penulis dapat meningkatkan ukurannya secara proaktif jika dirasa perlu, sehingga PindaiRasa selalu siap sedia untuk menampung data yang terus bertambah dan memberikan pengalaman terbaik bagi pengguna.

KESIMPULAN DAN SARAN

Aplikasi *mobile* PindaiRasa telah berhasil dibangun dengan sistem *back-end* menggunakan *framework* Node.js Express. Pengembangan *back-end* dilakukan dengan metode *Waterfall* yang terstruktur dan terukur, serta diimplementasikan menggunakan layanan *Google Cloud Platform* (GCP) dan Pengujian menyeluruh dengan metode *black box testing* membuktikan bahwa semua API yang dibuat berfungsi dengan baik dan sesuai dengan harapan. Kolaborasi erat antara tim *back-end* dan tim mobile *development* memastikan bahwa seluruh *endpoint* API terintegrasi dengan mulus dan siap digunakan.

DAFTAR PUSTAKA

A. Mubariz et al., "Perancangan *Back-end Server* Menggunakan Arsitektur Rest dan *Platform* Node.JS (Studi Kasus: Sistem Pendaftaran Ujian Masuk Politeknik Negeri Ujung Pandang),"2020.

B. R. Suteja and R. Agustaf, "Interoperabilitas Aplikasi Berbasis Web Service," Jurnal Informasi Interaktif, vol. 5, no. 3, pp. 106–114, 2020.

Hasanuddin, H. Asgar, and B. Hartono, "Rancang Bangun Rest Api Aplikasi Weshare Sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan," JINTEKS (Jurnal Informatika Teknologi dan Sains), vol. 4, no. 1, pp. 8–14, 2022.

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.

Liu, C., Yu, X., & Zhou, Z. (2014). The effectiveness of black-box testing techniques for Java applications. Software Quality Journal, 22(3), 593-614.

Ramadhan, Muhammad & Zukhri, Zainudin. (2023). "Pengembangan Rest Api Sistem Uii Admisi Dengan Menggunakan Pendekatan Domain Driven Design". Jurnal Ilmiah Informatika. 11. 176-182.

S. Ahmed and Q. Mahmood, "An authentication based scheme for applications using JSON web token," 2019 *22nd International Multitopic Conference (INMIC)*, Islamabad, Pakistan, 2019, pp.1-6.

S. Nugraha, A. B. Prasetijo, and D. Eridani, "Perancangan *Backend* Aplikasi Reservasi Talanoa Kopi and Space Menggunakan Framework Express.js Back End Design of the Talanoa Kopi and Space Reservation Application Using Express.js Framework," Jurnal Teknik Komputer, vol. 1, no. 3, pp. 126–131, 2022.

Santoso, A., & Wijaya, S. (2022). Pengembangan *Platform* IoT Cloud berbasis Layanan Komputasi *Server*less Google Cloud *Platform* (GCP). j-ptiik, 5(2), 4831-4842.

Setyowidi, D. A., & Anggoro, A. 'Membangun Aplikasi Web Skalabilitas Tinggi dengan Cloud Run dan Cloud SQL". Jurnal Rekursif Jurnal Rekursif, Vol. 5 No. 1, ISSN 2303-0755. 2023.

T. Bratakusuma, I. U. Azmi, and S. Ayuningtiyas, "Pengembangan Back End Pada Aplikasi Alat Tulis Kantor Bank Indonesia Perwakilan Purwokerto Menggunakan Nodejs Back End *Development* on Stationary Applications Bank Indonesia Representative Office Purwokerto Using Nodejs," 2022.

Uppal, S. Srivastava and K. Saini, "Web *Development* Framework : Future Trends," *2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, Greater Noida, India, 2022, pp. 2181-2184.

Yuliani, R., & Mardiah, F. (2022). Perbandingan Penerapan Metode *Waterfall* dan Agile pada Pengembangan Sistem Informasi Akademik. Jurnal Teknologi Informasi dan Komputer, 11(1), 1-10.

Perancangan Backend Api Berbasis Rest-Api pada Aplikasi Rekomendasi Resep Makanan

ORIGINALITY REPORT

21%	18%	7%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	jurnal.poliupg.ac.id Internet Source	4%
2	jurnal.untan.ac.id Internet Source	2%
3	journal.arteii.or.id Internet Source	2%
4	ojs.jurnaltechne.org Internet Source	1%
5	www.irjmets.com Internet Source	1%
6	Submitted to Universitas Terbuka Student Paper	1%
7	journal.aritekin.or.id Internet Source	1%
8	upc.aws.openrepository.com Internet Source	1%
9	ejournal.pnc.ac.id Internet Source	1%

10	repository.uksw.edu Internet Source	1 %
11	www.scribd.com Internet Source	1 %
12	Submitted to RMIT University Student Paper	1 %
13	Rama Sakti Hafidz Fadhilah Aziz, Irwan A. Kautsar, Sumarno. "Implementasi Domain Driven Design dan Clean Architecture dalam Pengembangan Web Service Aplikasi Alifarm Digital", Journal of Internet and Software Engineering, 2024 Publication	1 %
14	Submitted to Sriwijaya University Student Paper	<1 %
15	Submitted to Universitas Maritim Raja Ali Haji Student Paper	<1 %
16	dielektrika.unram.ac.id Internet Source	<1 %
17	j-ptiik.ub.ac.id Internet Source	<1 %
18	ojs.stmik-banjarbaru.ac.id Internet Source	<1 %
19	talenta.usu.ac.id Internet Source	<1 %

20	Rangga Gelar Guntara, Varinia Azkarin. "Implementasi dan Pengujian REST API Sistem Reservasi Ruang Rapat dengan Metode Black Box Testing", Jurnal Minfo Polgan, 2023 Publication	<1 %
21	digilib.uinsby.ac.id Internet Source	<1 %
22	karya-ilmiah.um.ac.id Internet Source	<1 %
23	www.duniailkom.com Internet Source	<1 %
24	repositori.uin-alauddin.ac.id Internet Source	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	Off		