

Pengembangan Application Programming Interface Berbasis REST API untuk Menampilkan Informasi Kegiatan Gereja

Immanuel Sebastian Wibisono¹, Noven Indra Prasetya²

¹Program Studi Informatika, Fakultas Teknik, Universitas Wijaya Kusuma Surabaya

²Program Studi Informatika, Fakultas Teknik, Universitas Wijaya Kusuma Surabaya

Email: ¹iman.sebastian88@gmail.com, ²noven@uwks.ac.id

Abstract. *This study aims to develop a REST-based Application Programming Interface (API) designed to display church activity information in real-time through the integration of the Church Information System (SIG) and the church website. The development methodology used is Waterfall, which includes needs analysis, design, implementation, and system testing. The needs analysis resulted in an API design with several endpoints for managing church activity data, such as worship schedules, inaugurations, and visitations. System testing was conducted based on the ISO 9126 standard, covering the aspects of functionality, usability, and efficiency. The test results indicate that the API achieved a functionality level of 100%, signifying that all features function according to the specified requirements. In terms of usability, the API obtained a score of 86%, categorized as "Good", indicating a high level of ease of use and seamless integration with external systems. Meanwhile, in terms of efficiency, the API demonstrated an average response time of 117.57 ms, reflecting optimal performance in handling data requests quickly and efficiently. The findings of this study indicate that the REST API not only enhances data management efficiency but also provides real-time information accessibility to the congregation. This study makes a significant contribution to the digitalization of religious institutions while also opening opportunities for further development, such as integration with mobile applications and enhancements in data security.*

Keywords: REST API, digitization, church information system, ISO 9126, data integration.

Abstrak. *Penelitian ini bertujuan untuk mengembangkan Application Programming Interface (API) berbasis REST yang dirancang untuk menampilkan informasi kegiatan gereja secara real-time melalui integrasi antara Sistem Informasi Gereja (SIG) dan website gereja. Metodologi pengembangan yang digunakan adalah Waterfall, yang meliputi analisis kebutuhan, desain, implementasi, dan pengujian sistem. Analisis kebutuhan menghasilkan rancangan API dengan beberapa endpoint untuk mengelola data kegiatan gereja, seperti jadwal kebaktian, pelantikan, dan visitasi. Pengujian sistem dilakukan berdasarkan standar ISO 9126, yang mencakup aspek functionality, usability, dan efficiency. Hasil pengujian menunjukkan bahwa API mencapai tingkat functionality sebesar 100%, yang mengindikasikan bahwa seluruh fitur telah berfungsi sesuai dengan spesifikasi yang ditetapkan. Dari aspek usability, API memperoleh skor 86%, yang masuk dalam kategori "Baik", menandakan bahwa API memiliki tingkat kemudahan penggunaan yang tinggi serta mendukung integrasi dengan sistem eksternal. Sementara itu, dari aspek efficiency, API menunjukkan waktu respons rata-rata 117,57 ms, yang mencerminkan kinerja optimal dalam menangani permintaan data secara cepat dan efisien. Hasil penelitian ini menunjukkan bahwa REST API tidak hanya meningkatkan efisiensi pengelolaan data, tetapi juga memberikan aksesibilitas informasi secara real-time kepada jemaat. Penelitian ini memberikan kontribusi signifikan terhadap digitalisasi institusi keagamaan, sekaligus membuka peluang pengembangan lebih lanjut, seperti integrasi dengan aplikasi mobile dan peningkatan keamanan data.*

Kata Kunci: REST API, digitalisasi, sistem informasi gereja, ISO 9126, integrasi data.

1. Pendahuluan

Dalam era transformasi digital, kebutuhan akan informasi yang akurat, terintegrasi, dan mudah diakses telah menjadi aspek yang sangat penting bagi berbagai jenis organisasi, termasuk institusi

keagamaan seperti gereja. Gereja, sebagai pusat kegiatan keagamaan, pendidikan, dan sosial, memegang peranan penting dalam menyampaikan informasi kepada komunitasnya secara efisien. Namun, keterbatasan dalam penggunaan metode konvensional seperti warta cetak, pengumuman lisan, dan media sosial yang tidak terstruktur sering kali menjadi hambatan dalam mencapai efisiensi tersebut. Di sinilah peran teknologi berbasis web, seperti *Application Programming Interface* (API), menjadi solusi potensial untuk mengatasi tantangan tersebut. *REST API* (*Representational State Transfer*) adalah salah satu pendekatan yang paling umum digunakan dalam pengembangan teknologi web karena kemampuannya untuk mengintegrasikan data dari berbagai sumber serta memberikan akses yang cepat dan mudah kepada pengguna melalui platform yang berbeda [1], [2].

REST API menawarkan fleksibilitas tinggi dalam pengelolaan data, memungkinkan organisasi untuk menyediakan layanan informasi secara *real-time* dengan efisiensi yang optimal. Dalam berbagai studi terdahulu, *REST API* telah terbukti meningkatkan aksesibilitas informasi pada berbagai domain, termasuk layanan kesehatan, pendidikan, dan pemerintahan [3], [4]. Namun, penerapan teknologi *REST API* dalam konteks institusi keagamaan, khususnya gereja, masih jarang dibahas secara mendalam. Kondisi ini menunjukkan adanya celah dalam literatur yang memerlukan perhatian lebih lanjut untuk mengembangkan sistem yang mampu mendukung penyebaran informasi kegiatan gereja secara lebih terstruktur dan responsif.

Banyak gereja masih mengandalkan metode komunikasi yang bersifat satu arah dan kurang adaptif terhadap kebutuhan masyarakat digital saat ini. Metode ini sering kali tidak mencakup aspek *real-time*, integrasi data, serta kemudahan akses oleh komunitas yang lebih luas, termasuk mereka yang berada di luar jangkauan fisik gereja [5], [6]. Masalah ini tidak hanya memengaruhi efektivitas komunikasi internal, tetapi juga membatasi peluang gereja untuk memperluas keterlibatan komunitas melalui teknologi digital yang lebih modern.

Penelitian terdahulu menunjukkan bahwa *REST API* dapat menjadi solusi ideal untuk mengatasi permasalahan ini. Misalnya, *REST API* telah digunakan untuk mengintegrasikan sistem informasi akademik, layanan kesehatan, dan pemerintahan, sehingga memungkinkan pengguna untuk mengakses informasi dengan cepat dan mudah melalui berbagai perangkat [7], [8]. Dalam konteks gereja, *REST API* dapat dirancang untuk mengintegrasikan data dari berbagai sumber, seperti jadwal kebaktian, informasi pelayanan, dan acara khusus, ke dalam satu platform yang terstruktur [9], [10]. Meskipun demikian, tantangan yang muncul adalah bagaimana merancang *REST API* yang tidak hanya memenuhi kebutuhan fungsional tetapi juga mampu menangani kompleksitas spesifik dari institusi keagamaan.

Salah satu celah utama dalam literatur adalah kurangnya penelitian tentang bagaimana *REST API* dapat dioptimalkan untuk konteks gereja, yang memiliki karakteristik unik seperti kebutuhan akan keamanan data, integrasi dengan sistem pelayanan internal, dan aksesibilitas bagi pengguna dengan tingkat literasi digital yang beragam [11], [12]. Oleh karena itu, penelitian ini bertujuan untuk menjembatani kesenjangan tersebut dengan mengembangkan *REST API* yang disesuaikan dengan kebutuhan spesifik komunitas gereja, sekaligus mempertimbangkan aspek efisiensi, keandalan, dan kemudahan penggunaan.

Penelitian ini bertujuan untuk mengembangkan *REST API* berbasis web yang berfungsi sebagai penghubung utama antara sistem internal gereja (SIG) dan *website* yang digunakan oleh jemaat. Tujuan utama penelitian ini adalah memastikan data kegiatan gereja dapat dikelola dan ditampilkan secara *real-time* melalui proses permintaan dan pengembalian data yang efisien dan terstruktur. Kebaruan penelitian ini terletak pada pendekatan arsitektur yang berfokus pada pemisahan tanggung jawab (*separation of concerns*), memungkinkan integrasi yang lebih baik antara sistem internal gereja dengan *website* gereja, serta mendukung pengalaman pengguna (*user experience*) dengan berbagai tingkat literasi digital. Ruang lingkup penelitian mencakup analisis kebutuhan pengguna, perancangan arsitektur *REST API*, implementasi sistem, dan evaluasi kinerja berdasarkan parameter seperti efisiensi, keandalan, dan aksesibilitas. Evaluasi dilakukan menggunakan metode *Black Box Testing* serta penilaian kualitas perangkat lunak berdasarkan standar ISO 9126, yang mencakup *functionality*, *usability*, dan *efficiency*.

Hasil pengujian menunjukkan bahwa API yang dikembangkan telah berfungsi sesuai dengan spesifikasi yang ditetapkan (*functionality*), memiliki tingkat kemudahan penggunaan yang baik (*usability*), serta mampu merespons permintaan data dengan cepat dan akurat (*efficiency*). Hasil dari

penelitian ini diharapkan dapat memberikan kontribusi baik secara teoritis maupun praktis, serta juga menjadi model integrasi data yang dapat diadopsi oleh institusi keagamaan lain dengan kebutuhan serupa.

2. Tinjauan Pustaka

2.1 Application Programming Interface (API)

Application Programming Interface (API) adalah sekumpulan aturan, definisi, dan protokol yang dirancang untuk memfasilitasi interaksi antara berbagai aplikasi perangkat lunak [13]. API memungkinkan pengembang untuk membangun sistem yang dapat saling berkomunikasi secara efisien, tanpa harus memahami sepenuhnya struktur internal dari aplikasi yang terhubung. Dengan API, aplikasi dapat berbagi data dan fungsi dengan mudah, menciptakan ekosistem yang terintegrasi.

Selain itu, API memainkan peran penting dalam pengembangan teknologi modern dengan memungkinkan integrasi lintas platform. Misalnya, API sering digunakan untuk menghubungkan aplikasi dengan layanan pihak ketiga, seperti pembayaran *online*, pengiriman data, atau integrasi media sosial. Dengan adanya API, pengembangan aplikasi menjadi lebih fleksibel, efisien, dan memungkinkan sistem yang terhubung untuk bekerja secara harmonis.

2.2 Arsitektur Web Service

Web Service adalah mekanisme komunikasi yang memungkinkan sistem komputer yang berbeda untuk berinteraksi satu sama lain [14]. Dengan memanfaatkan protokol standar seperti *HTTP* dan format data seperti XML atau JSON, Web Service memungkinkan aplikasi untuk mengakses fungsionalitas aplikasi lain tanpa perlu mengetahui detail implementasinya. Teknologi ini mendukung protokol seperti SOAP, yang menggunakan format XML, atau REST, yang lebih ringan dengan format JSON. Keunggulan utama Web Service terletak pada fleksibilitas dan interoperabilitasnya, memungkinkan pengembang untuk mengintegrasikan berbagai komponen dan aplikasi secara modular, mempercepat pengembangan, serta memudahkan pemeliharaan dan pembaruan sistem. Selain itu, Web Service mendukung komunikasi sinkron dan asinkron, sehingga dapat disesuaikan dengan kebutuhan aplikasi. Dengan fleksibilitas dan skalabilitasnya, Web Service menjadi fondasi penting dalam pengembangan aplikasi modern dan meningkatkan efisiensi operasional.

2.3 REST API

REST API adalah jenis arsitektur yang digunakan dalam *Application Programming Interface* (API) [14]. *REST API* memanfaatkan protokol *HTTP* untuk memungkinkan sistem saling berinteraksi dan bertukar data secara terstruktur dan standar. *REST API* sederhana dan efektif karena menggunakan metode *HTTP* standar seperti GET, POST, PUT, dan DELETE untuk mengelola sumber daya, serta mendukung format data ringan seperti JSON yang lebih cepat dan efisien dibanding XML. *REST API* bersifat *stateless*, sehingga setiap permintaan mencakup semua informasi yang diperlukan, membuatnya mudah diskalakan dan dikelola, terutama dalam aplikasi web dan *mobile modern*. Selain mendukung interoperabilitas antar sistem untuk menciptakan ekosistem teknologi yang terintegrasi, *REST API* juga memudahkan pengembangan dan pemeliharaan karena mengikuti prinsip yang terstandarisasi. Hal ini memungkinkan pengembang untuk memahami, menggunakan, dan memodifikasi API dengan cepat tanpa mengganggu sistem yang ada. Secara keseluruhan, *REST API* adalah solusi efisien untuk membangun aplikasi yang fleksibel, responsif, dan mampu berkomunikasi dengan berbagai layanan, menjadikannya standar utama dalam pengembangan layanan web saat ini.

3. Metode Penelitian

Tahap pengembangan sistem ini menggunakan metodologi *Waterfall*. Metode *Waterfall* memastikan bahwa setiap langkah dalam proses pengembangan dilakukan dengan urutan yang jelas dan terstruktur [15]. Metode ini diadopsi untuk memastikan setiap langkah pengembangan dilakukan secara menyeluruh. Metode ini terdiri dari analisis kebutuhan, desain, implementasi, dan pengujian.

3.1 Analisis Kebutuhan

Penulis melakukan analisis kebutuhan sebagai dasar untuk merancang sistem *RESTful* API yang sesuai dengan kebutuhan gereja. Analisis ini mencakup pemahaman terhadap skema data sistem internal gereja, termasuk struktur *database*, pola penyimpanan data, dan identifikasi *endpoint* yang diperlukan untuk mendukung pertukaran data secara efisien. Hasil dari analisis ini adalah merumuskan kebutuhan yang menjadi acuan dalam pengembangan sistem, yaitu :

- a. Data kegiatan dapat muncul secara *real time*
- b. Jemaat dapat mengakses informasi kegiatan melalui berbagai perangkat
- c. Informasi ditampilkan dengan jelas dan mudah dipahami
- d. Data yang telah dibuat dapat segera terupdate di *Website*

3.2 Desain

Tahap desain sistem bertujuan untuk memetakan alur kerja dan mendeskripsikan integrasi antara komponen-komponen utama yang menjadi bagian dari sistem yang akan dikembangkan. Pada tahap ini, dilakukan perancangan arsitektur sistem berbasis *separation of concerns* untuk memastikan bahwa data dari Sistem Informasi Gereja (SIG) dapat terintegrasi secara otomatis dengan situs web gereja menggunakan teknologi *REST API*.

3.3 Implementasi

Tahap ini diawali dengan memanfaatkan desain sistem yang telah dibuat sebagai acuan utama. Selanjutnya, pengembangan *REST API* dilakukan menggunakan *framework CodeIgniter* (CI), dengan pengelolaan basis data melalui SQL. *Output* dari tahap ini adalah *REST API* yang berperan sebagai penghubung antara sistem internal gereja dan situs web gereja, sehingga memungkinkan pertukaran data yang terstruktur dan aman.

3.4 Pengujian

Setelah tahap implementasi sistem selesai, dilakukan pengujian untuk menilai kualitas sistem berdasarkan standar ISO 9126. Pengujian ini mencakup tiga aspek utama, yaitu *functionality*, yang dievaluasi menggunakan metode *Black Box Testing* guna memastikan setiap fitur berfungsi sesuai dengan spesifikasi yang telah ditetapkan; *usability*, yang diukur melalui kuesioner berbasis skala Likert untuk menilai tingkat kemudahan penggunaan sistem; serta *efficiency*, yang dianalisis dengan bantuan *Postman* untuk mengukur waktu respons API. Hasil dari pengujian ini memberikan gambaran komprehensif mengenai performa sistem dan menjadi dasar untuk melakukan perbaikan serta optimisasi sebelum sistem diterapkan secara penuh.

4. Hasil dan Pembahasan

4.1 Analisis Kebutuhan

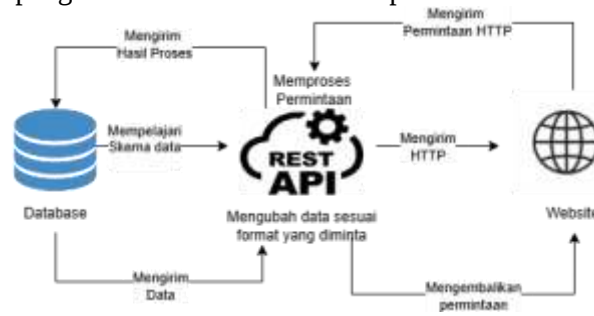
Berdasarkan hasil analisis kebutuhan, penulis mengidentifikasi bahwa sistem harus memiliki kemampuan untuk mengelola dan menyajikan data terkait berbagai kegiatan gereja secara terstruktur. Sistem ini dirancang untuk menjalankan dua fungsi utama: pertama, menampilkan daftar lengkap kegiatan gereja secara umum, dan kedua, menyediakan informasi rinci mengenai kegiatan tertentu sesuai dengan kebutuhan jemaat. Selain itu, analisis ini menekankan pentingnya pengelolaan data secara *real-time* untuk memastikan bahwa informasi yang disampaikan kepada jemaat selalu akurat dan mutakhir.

Dalam mendukung fungsi ini, penulis merancang beberapa *endpoint* untuk mengelola data kegiatan gereja, yaitu:

- a. Daftar Kegiatan: Menyajikan semua informasi kegiatan gereja dalam satu daftar kalender.
- b. Jadwal Kegiatan: Memberikan informasi rutin mengenai kegiatan bulanan.
- c. Baptis Sidi: Menyediakan data terkait pelaksanaan baptis dan sidi.
- d. Pernikahan: Menampilkan jadwal dan detail pasangan yang akan menikah.
- e. Kedukaan: Memberikan informasi mengenai ibadah kedukaan.
- f. Pelantikan: Menyajikan jadwal pelantikan majelis gereja.
- g. Visitasi: Memberikan informasi kunjungan jemaat atau pelayanan tertentu.

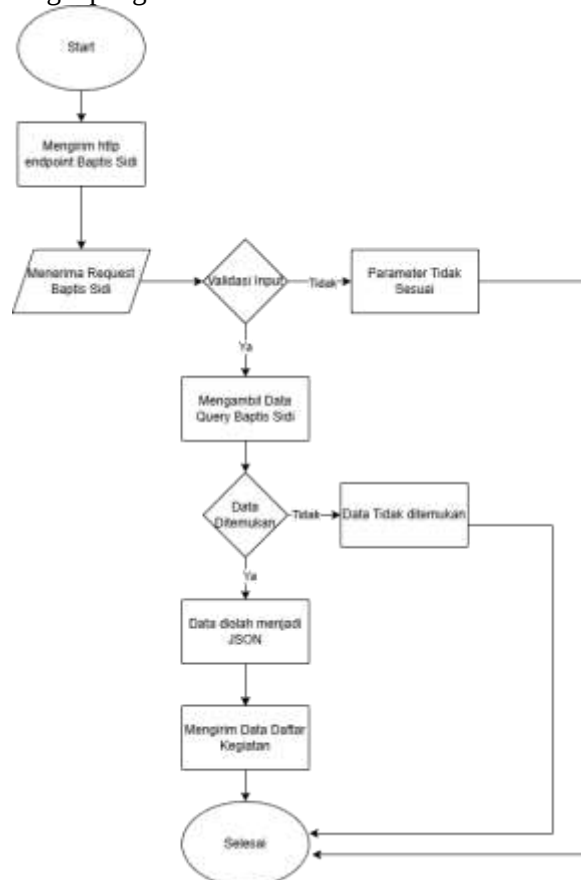
4.2 Desain

Desain ini melibatkan perancangan *REST API* yang berfungsi sebagai perantara untuk mentransfer data antara Sistem Informasi Gereja (SIG) dan situs web gereja. Tahap awal proses ini dimulai dengan menganalisis skema basis data pada SIG, meliputi struktur tabel, hubungan antar tabel, serta jenis data yang digunakan. Setelah skema data dipahami secara menyeluruh, *REST API* dirancang untuk menyediakan *endpoint HTTP* yang mendukung operasi berbasis metode *POST*, seperti pengiriman data baru dan pengambilan informasi melalui permintaan *POST*.



Gambar 1. Arsitektur Sistem

Proses kerja perancangan sistem dijelaskan pada Gambar 1. Dalam mekanisme ini, situs web gereja mengirimkan permintaan *HTTP* ke *endpoint REST API* yang telah dirancang secara khusus. *REST API* bertanggung jawab untuk menerima permintaan tersebut, melakukan pemrosesan, dan menerjemahkan tindakan yang diperlukan sesuai dengan jenis permintaan yang diterima. Selanjutnya, *REST API* mengakses basis data untuk memperoleh data yang diminta. Data hasil operasi pada basis data tersebut kemudian dikembalikan oleh *REST API* dalam format *JSON*, yang berfungsi sebagai sumber data bagi situs web gereja. Dengan rancangan ini, *REST API* tidak hanya berperan sebagai penghubung, tetapi juga sebagai pengelola alur data antara basis data dan situs web gereja.



Gambar 2. Flowchart Endpoint Baptis Sidi

Alur proses untuk endpoint Baptis Sidi dimulai ketika pengguna mengirimkan permintaan HTTP menggunakan metode POST ke endpoint Baptis Sidi, sebagaimana ditunjukkan pada Gambar 2. Setelah server menerima permintaan tersebut, sistem akan melakukan validasi terhadap parameter yang disertakan untuk memastikan kelengkapannya dan kesesuaiannya dengan format yang telah ditentukan. Jika validasi tidak berhasil, sistem akan mengembalikan respons berupa pesan kesalahan "Parameter Tidak Sesuai" kepada pengguna.

Apabila parameter yang diterima dinyatakan valid, sistem akan melanjutkan proses dengan melakukan pengambilan data Baptis Sidi dari basis data internal berdasarkan parameter yang telah diberikan. Jika data yang dimaksud tidak ditemukan dalam basis data, sistem akan memberikan respons berupa pesan "Data Tidak Ditemukan."

Sebaliknya, apabila data berhasil ditemukan, sistem akan mengolahnya menjadi format JSON yang mudah diakses dan dikirimkan kepada pengguna. Setelah proses pengolahan selesai, sistem akan mengirimkan respons yang berisi data Baptis Sidi sesuai permintaan pengguna.

4.3 Implementasi

Pada tahap implementasi, sistem menghasilkan sebuah situs web yang dirancang untuk menyajikan informasi kegiatan gereja secara real-time melalui integrasi dengan *RESTful API*. Sebagaimana ditunjukkan pada Gambar 3, tampilan awal antarmuka pada situs web gereja memperlihatkan desain sebelum data kegiatan diintegrasikan melalui Sistem Informasi Gereja (SIG) menggunakan *REST API*. Pada sisi kiri halaman, terdapat kalender digital yang menampilkan hari dan tanggal dalam satu bulan, sedangkan di sisi kanan disediakan ruang untuk menampilkan daftar kegiatan lengkap dengan tanggal dan waktu pelaksanaannya. Tampilan awal ini masih berupa antarmuka kosong yang dirancang untuk menerima data dari SIG, sehingga nantinya dapat menyediakan informasi yang relevan dan terkini bagi pengguna.



Gambar 3. Portal Website

Pengambilan data dilakukan dengan menggunakan *query SQL* yang menggabungkan beberapa tabel melalui teknik *JOIN*, *CONCAT*, dan *GROUP_CONCAT*. Data yang relevan diambil berdasarkan parameter input yang dikirimkan oleh klien. Pada sisi *backend*, API memastikan validitas data melalui proses validasi, kemudian menyusun data tersebut dalam format array terstruktur sebelum dikonversi ke format JSON menggunakan fungsi *json_encode*. Hasil akhir dari integrasi ini ditampilkan pada Gambar 4, di mana informasi kegiatan gereja dapat disajikan secara dinamis dalam kalender, memberikan akses informasi yang terkini dan efisien kepada pengguna.



Gambar 4. Hasil Implementasi

4.4 Pengujian

4.4.1. *Functionality*

Berdasarkan pengujian *functionality* menggunakan metode *Black Box Testing*, diperoleh hasil bahwa seluruh 10 skenario uji berhasil (sukses).

Tabel 1. Hasil pengujian *functionality*

No	Endpoint	Input	Output	Result
1.	daftar_kegiatan	Memasukkan tanggal awal "2024-11-01" dan tanggal akhir "2024-11-30"	Menampilkan semua daftar kegiatan dari rentang tanggal tersebut	Sukses
2.	daftar_kegiatan	Memasukkan tanggal awal kosong dan tanggal akhir "2024-11-30"	Menampilkan pesan "Data Tidak Ditemukan"	Sukses
3.	jadwal_kegiatan	Memasukkan kode "JKG00002"	Menampilkan Data kegiatan untuk kode tersebut	Sukses
4.	baptis_sidi	Memasukkan kode "BPT00001"	Menampilkan Data baptis atau sidi untuk kode tersebut	Sukses
5.	kedukaan	Memasukkan kode "KD00001"	Menampilkan Data kedukaan untuk kode tersebut	Sukses
6.	pernikahan	Memasukkan kode "PN00001"	Menampilkan Data pernikahan untuk kode tersebut	Sukses
7.	pelantikan	Memasukkan kode "PJG00002"	Menampilkan Data pelantikan untuk kode tersebut	Sukses
8.	visitasi	Memasukkan kode "VSS00001"	Menampilkan Data visitasi untuk kode tersebut	Sukses
9.	visitasi	Memasukkan kode "VSS00001" dan nomor "1"	Menampilkan Data visitasi untuk kode tersebut	Sukses
10.	visitasi	Memasukkan nomor "1"	Menampilkan Data visitasi untuk kode tersebut	Sukses

Nilai fungsionalitas perangkat lunak dapat dihitung menggunakan rumus berikut:

$$Functionality\ Score = 1 - \frac{Total\ Skenario\ Gagal}{Total\ Skenario\ Uji} = 1 - \frac{0}{10} = 1 \quad (1)$$

Berdasarkan hasil perhitungan pengujian *functionality*, diperoleh nilai 1, yang menunjukkan bahwa kualitas API yang dikembangkan tergolong baik. Hal ini sesuai dengan interpretasi ISO 9126, di mana nilai *functionality* yang optimal adalah yang mendekati 1 dalam rentang $0 \leq X \leq 1$ [16]. Dengan hasil tersebut, dapat disimpulkan bahwa REST API telah memenuhi standar *functionality* yang ditetapkan dan layak untuk digunakan.

4.4.2. *Usability*

Pengujian *usability* dilakukan menggunakan kuesioner yang berisi pertanyaan terkait penggunaan REST API, dengan melibatkan 10 responden yang merupakan mahasiswa tingkat akhir Program Studi Teknik Informatika.

Tabel 2. Hasil pengujian *Usability*

No	Pertanyaan	SS	S	N	TS	KS
1.	API menampilkan jadwal kegiatan gereja secara <i>real-time</i>	3	4	2	1	0
2.	API mempermudah integrasi antara Sistem Informasi Gereja (SIG) dan website gereja.	3	5	2	0	0
3.	Endpoint API mudah digunakan untuk mengambil data jadwal kegiatan gereja	2	4	3	1	0
4.	Pesan <i>error</i> yang diberikan API sudah jelas dan mudah dipahami	2	4	2	1	1
5.	API memberikan respons yang cepat dan akurat terhadap permintaan data.	2	5	2	1	0

Dari skor *usability* mendapatkan data sebagai berikut

$$Usability\ Score = \frac{Total\ Skor\ yang\ Diperoleh}{Total\ Skor\ Maksimal} \times 100\% = \frac{215}{250} \times 100\% = 86\% \quad (2)$$

Berdasarkan hasil pengujian, API Sistem Informasi Gereja memperoleh tingkat *usability* sebesar 86%, yang dikategorikan dalam skala "Baik" sesuai dengan standar evaluasi kualitas perangkat lunak [16].

4.4.3. Efficiency

Tabel 3. Hasil pengujian Efficiency

No	Endpoint	Respon Time
1.	daftar_kegiatan	222 ms
2.	jadwal_kegiatan	239 ms
3.	baptis_sidi	111 ms
4.	kedukaan	60 ms
5.	pernikahan	70 ms
6.	pelantikan	61 ms
7.	visitasi	60 ms
	Rata-rata	117,57 ms

Berdasarkan pengujian efisiensi menggunakan *Postman*, hasil yang disajikan pada Tabel 3 menunjukkan bahwa rata-rata waktu respons API mencapai 117,57 ms. Dengan nilai tersebut, sistem dapat dikategorikan memiliki kinerja optimal dalam menangani permintaan data secara cepat dan efisien, sehingga mendukung efektivitas dalam proses integrasi dan pengelolaan informasi [17].

5. Kesimpulan

Penelitian ini telah berhasil mengembangkan *Application Programming Interface* (API) berbasis REST untuk menampilkan informasi kegiatan gereja secara *real-time*. Hasil penelitian menunjukkan bahwa penerapan REST API berkontribusi dalam meningkatkan efisiensi, akurasi, dan aksesibilitas dalam pengelolaan data kegiatan gereja. Pendekatan arsitektur berbasis *separation of concerns* memungkinkan integrasi data antara Sistem Informasi Gereja (SIG) dan website gereja secara lebih optimal.

Berdasarkan hasil pengujian menggunakan ISO 9126, API yang dikembangkan memperoleh hasil sebagai berikut:

- Functionality*, dengan tingkat keberhasilan 100%, menunjukkan bahwa seluruh fitur API telah berfungsi dengan baik sesuai spesifikasi yang ditetapkan.
- Usability* mencapai 86%, yang dikategorikan dalam skala "Baik", menunjukkan bahwa API memiliki tingkat kemudahan penggunaan yang tinggi serta mendukung integrasi dengan sistem eksternal.
- Efficiency*, dengan rata-rata waktu respons 117,57 ms, mengindikasikan bahwa API memiliki kinerja yang optimal dalam menangani permintaan data dengan cepat dan efisien.

Hasil penelitian ini menegaskan bahwa REST API merupakan solusi teknologi yang efektif dalam mendukung digitalisasi institusi keagamaan. Secara khusus, penelitian ini memberikan kontribusi dengan menghadirkan model arsitektur API yang dapat diterapkan pada institusi keagamaan lainnya, baik dalam konteks teoritis maupun praktis.

Penelitian ini juga membuka peluang untuk pengembangan lebih lanjut, seperti integrasi REST API dengan aplikasi *mobile*, pengelolaan data yang lebih kompleks, serta penerapannya pada institusi dengan kebutuhan serupa. Selain itu, penelitian masa depan dapat difokuskan pada evaluasi keamanan data serta peningkatan pengalaman pengguna, khususnya bagi komunitas dengan literasi digital yang beragam.

Secara keseluruhan, penelitian ini menegaskan bahwa penerapan REST API tidak hanya meningkatkan efisiensi pengelolaan data, tetapi juga berperan dalam mendukung transformasi digital pada institusi keagamaan. Dengan demikian, hasil penelitian ini menjadi dasar yang kuat bagi pengembangan teknologi informasi yang lebih inovatif di masa depan.

Referensi

- [1] G. Jones *et al.*, "API Development Increases Access to Shared Computing Resources at Boston University," *J. Softw. Eng. Appl.*, vol. 15, no. 06, pp. 197–207, 2022, doi: 10.4236/jsea.2022.156011.
- [2] Ach. Khozaimi, Y. D. Pramudita, and F. Solihin, "DESIGN AND DEVELOPMENT OF BACKEND APPLICATION FOR THESIS MANAGEMENT SYSTEM USING MICROSERVICE ARCHITECTURE AND RESTFUL API," *J. Ilm. Kursor*, vol. 11, no. 4, pp. 179–186, Jan. 2023, doi: 10.21107/kursor.v11i4.313.
- [3] O. Baniş, D. Florea, R. Gyalai, and D.-I. Curiac, "Automated Specification-Based Testing of REST APIs," *Sensors*, vol. 21, no. 16, p. 5375, Aug. 2021, doi: 10.3390/s21165375.
- [4] J. Bogner, S. Kotstein, and T. Pfaff, "Do RESTful API design rules have an impact on the understandability of Web APIs?," *Empir. Softw. Eng.*, vol. 28, no. 6, p. 132, Nov. 2023, doi: 10.1007/s10664-023-10367-y.
- [5] V. Azkarin, R. G. Guntara, and O. Herdiana, "Development of a REST API for Human Resource Information System for Employee Referral Management Domain Using the Express JS Framework and Node.js," *J. Sci. Res. Educ. Technol. JSRET*, vol. 2, no. 3, pp. 1085–1094, Jul. 2023, doi: 10.58526/jsret.v2i3.199.
- [6] A. A. Prayogi, M. Niswar, Indrabayu, and M. Rijal, "Design and Implementation of REST API for Academic Information System," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 875, no. 1, p. 012047, Jun. 2020, doi: 10.1088/1757-899X/875/1/012047.
- [7] T. Anjarsari and F. Ardiani, "Application of Rest Api Technology in Android-Based Beauty Salon Service Reservation System," *J. Comput. Sci. Technol. Stud.*, vol. 5, no. 4, pp. 203–212, Dec. 2023, doi: 10.32996/jcsts.2023.5.4.21.
- [8] N. A. Prasetyo and Y. Saintika, "Integration between Moodle and Academic Information System using Restful API for Online Learning," *J. Ilm. Tek. Elektro Komput. Dan Inform.*, vol. 7, no. 2, p. 358, Oct. 2021, doi: 10.26555/jiteki.v7i2.21816.
- [9] M. C. Gursesli *et al.*, "Design of Cloud-Based Real-Time Eye-Tracking Monitoring and Storage System," *Algorithms*, vol. 16, no. 7, p. 355, Jul. 2023, doi: 10.3390/a16070355.
- [10] D. Prasetyawan and P. D. Rahmanto, "Pengembangan Sistem Seleksi Proposal Penelitian Berbasis Web Service Menggunakan REST API," *JTIM J. Teknol. Inf. Dan Multimed.*, vol. 6, no. 3, pp. 283–295, Sep. 2024, doi: 10.35746/jtim.v6i3.585.
- [11] A. C. Ningtyas and N. I. Prasetya, "SISTEM PENDUKUNG KEPUTUSAN PENERIMAAN BEASISWA PADA SMA YPI DARUSSALAM CERME MENGGUNAKAN METODE SIMPLE ADDITIVE WEIGHTING," *Melek IT Inf. Technol. J.*, vol. 8, no. 2, Jun. 2024, doi: 10.30742/melekitjournal.v8i2.216.
- [12] I. M. Sukarsa, I. K. A. M. Antara, P. W. Buana, I. P. A. Bayupati, N. W. Wisswani, and D. W. Puteri, "Data storage model in low-cost mobile applications," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 28, no. 2, p. 1128, Nov. 2022, doi: 10.11591/ijeecs.v28.i2.pp1128-1138.
- [13] H. Setiawan and M. N. Ghiffari, "Sistem Informasi Covid-19 Berbasis Mobile Dengan Framework Flutter dan Application Programming Interface (API)," *J. Teknol. Inf. Dan Komun. TIKomSiN*, vol. 10, no. 2, p. 35, Nov. 2022, doi: 10.30646/tikomsin.v10i2.640.
- [14] I. A. K. P. Paramitha, D. M. Wiharta, and I. M. Arsa Suyadnya, "PERANCANGAN DAN IMPLEMENTASI RESTFUL API PADA SISTEM INFORMASI MANAJEMEN DOSEN UNIVERSITAS UDAYANA," *J. SPEKTRUM*, vol. 9, no. 3, p. 15, Sep. 2022, doi: 10.24843/SPEKTRUM.2022.v09.i03.p3.
- [15] F. Tuakia and N. I. Prasetya, "RANCANG BANGUN APLIKASI PENCARIAN LOKASI OBJEK WISATA DESA SUKO BERBASIS ANDROID," *Melek IT Inf. Technol. J.*, vol. 7, no. 2, pp. 13–22, Dec. 2021, doi: 10.30742/melekitjournal.v7i2.166.

- [16] D. Gunawan and A. Kusumastuti, "Pengembangan dan Analisis Kualitas Sistem Informasi Manajemen Alumni Berbasis Website Menggunakan ISO 9126," *Indones. J. Comput. Sci.*, vol. 12, no. 2, Apr. 2023, doi: 10.33022/ijcs.v12i2.3191.
- [17] F. N. Diansah and U. Zaky, "PERBANDINGAN KECEPATAN PENGAMBILAN DATA ANTARA API PUBLIK AN MONGODB DALAM APLIKASI AGREGASI BERITA," .
COSTING.