Web-Based Library Collection Information System Using an API-Driven Architecture with the VEF Stack
[Sistem Informasi Koleksi Pustaka Berbasis Web Dengan Metode API-Driven Berbasis Stack VEF]

Riski Putra Alamzah [1)], Nuril Lutvi Azizah [*,2)] , Ika Ratna Indra Astutik[3)], Hamzah Setiawan[4)]

[1,2,3,4)]Program Studi Informatika, Universitas Muhammadiyah Sidoarjo, Indonesia
*Email Penulis Korespondensi: nurillutviazizah@umsida.ac.id

Abstract. *Digital libraries require systems that provide collection access, academic work distribution, transaction control, and structured validation. This study designs and develops a web-based digital library collection reservation and distribution information system using an API-driven architecture with the VEF stack. The system was developed with Nuxt 3/Vue 3 on the frontend, Express.js as the backend API, Firebase Firestore as a NoSQL database, Google Drive API for file storage, and AI-assisted modules for PDF metadata analysis, publication-source detection, and cover generation. The development method used Agile Scrum through problem identification, data collection, product backlog preparation, sprint planning, sprint backlog preparation, sprint development, sprint review and retrospective, implementation, and testing. The implemented system supports catalogue management, bookmarking or reservation, academic work submission, collection verification, external-source references, manual payment transactions, download history, and academic payout management. Black Box Testing on 20 scenarios showed that all main features worked as expected, while User Acceptance Testing involving 20 respondents obtained an acceptance score of 98.75%, categorized as highly feasible.*

Keywords - *AI-assisted metadata; API-driven; digital library; Firebase Firestore; Nuxt 3; reservation.*

Abstrak. *Perpustakaan digital membutuhkan sistem yang tidak hanya menyediakan akses koleksi, tetapi juga mendukung distribusi karya akademik, kontrol transaksi, dan validasi akses secara terstruktur. Penelitian ini bertujuan merancang dan membangun sistem informasi reservasi dan distribusi koleksi pustaka digital berbasis web menggunakan arsitektur API-driven dengan stack VEF. Sistem dikembangkan menggunakan Nuxt 3/Vue 3 pada sisi frontend, Express.js sebagai backend API, Firebase Firestore sebagai basis data NoSQL, Google Drive API sebagai penyimpanan dokumen, serta modul AI untuk analisis metadata PDF, deteksi indikasi sumber publikasi, dan pembuatan cover koleksi. Metode pengembangan yang digunakan adalah Agile Scrum melalui tahapan identifikasi masalah, pengumpulan data, penyusunan product backlog, sprint planning, penyusunan sprint backlog, sprint development, sprint review dan retrospective, implementasi, dan pengujian. Hasil implementasi menunjukkan bahwa sistem mampu mengelola katalog, reservasi atau bookmarking, unggah karya akademisi, verifikasi koleksi, rujukan sumber eksternal, transaksi pembayaran manual, riwayat unduhan, dan payout akademisi. Pengujian Black Box terhadap 20 skenario menunjukkan seluruh fitur utama berjalan sesuai harapan, sedangkan User Acceptance Testing terhadap 20 responden memperoleh penerimaan sebesar 98,75% dan termasuk kategori sangat layak.*

Kata Kunci - *AI-assisted metadata; API-driven; Firebase Firestore; koleksi pustaka digital; Nuxt 3; reservasi.*

I. PENDAHULUAN

Perpustakaan memiliki peran penting dalam mendukung proses pendidikan tinggi karena menjadi pusat penyedia informasi ilmiah. Perkembangan teknologi informasi mendorong perubahan perilaku pengguna dari peminjaman fisik menuju akses pustaka digital. Koleksi elektronik dianggap lebih praktis karena dapat diakses dari berbagai tempat dan membantu mahasiswa memperoleh sumber rujukan secara cepat [1][2]. Meskipun demikian, beberapa sistem perpustakaan digital masih memiliki keterbatasan dari sisi personalisasi, distribusi karya akademik, integrasi layanan berbasis cloud, dan pengendalian akses koleksi berbayar [3][4].

Permasalahan lain yang muncul adalah distribusi karya akademik dari dosen, peneliti, maupun praktisi sering kali belum terkelola dalam satu platform terbuka. Sistem yang sudah ada umumnya berfokus pada penyediaan koleksi internal dan belum memberikan ruang yang cukup bagi akademisi eksternal untuk mengunggah karya atau mengarahkan pengguna ke sumber publikasi asli secara tertib. Di sisi pengguna, fitur penyimpanan koleksi atau bookmarking juga belum selalu tersedia, padahal fitur tersebut membantu pengguna menandai koleksi yang relevan untuk dibaca kembali [5][6].

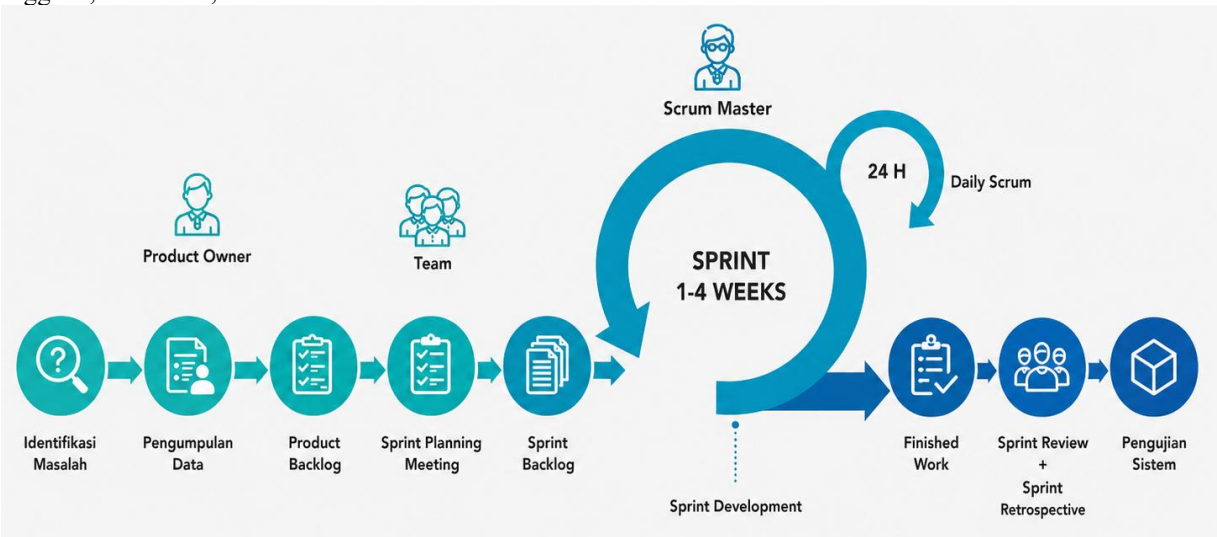
Kesenjangan tersebut menunjukkan perlunya sistem yang menggabungkan katalog digital, reservasi personal, distribusi koleksi akademik, validasi kontributor, dan kontrol akses unduhan. Penelitian ini menawarkan sistem informasi reservasi dan distribusi koleksi pustaka digital berbasis web. Sistem menyediakan akses koleksi gratis, mendukung koleksi berbayar dengan verifikasi transaksi manual, menyediakan rujukan sumber eksternal untuk karya yang telah terbit pada platform lain, serta menerapkan verifikasi akademisi agar kontributor telah melalui proses validasi oleh admin.

Arsitektur yang digunakan adalah API-driven dengan stack VEF, yaitu Vue.js pada sisi antarmuka, Express.js sebagai backend API, dan Firebase Firestore sebagai basis data. Pada implementasinya, Vue.js diterapkan melalui Nuxt 3 untuk mendukung routing, layout, middleware, dan state management. Backend Express.js berperan mengelola autentikasi, otorisasi, koleksi, reservasi, transaksi, verifikasi, integrasi eksternal, fitur AI, dan payout. Pendekatan API-driven dipilih karena memisahkan frontend dan backend sehingga sistem lebih modular, mudah diuji, dan dapat dikembangkan untuk integrasi platform lain [7][9].

Penelitian terkait menunjukkan bahwa sistem perpustakaan digital dapat meningkatkan efisiensi pengelolaan koleksi, tetapi sebagian besar masih berfokus pada pencarian dan pengunduhan langsung [6][10]. Kajian lain memperlihatkan bahwa Firebase dapat digunakan sebagai basis data berbasis cloud, sedangkan pengembangan sistem informasi dengan metode Agile maupun Scrum membantu proses pengembangan menjadi lebih adaptif terhadap kebutuhan pengguna [11][19]. Tujuan penelitian ini adalah membangun sistem informasi reservasi dan distribusi koleksi pustaka digital yang mendukung akses terbuka dan berbayar, menerapkan arsitektur API-driven, menyediakan mekanisme verifikasi akademisi, verifikasi koleksi, verifikasi transaksi pembayaran manual, serta menambahkan bantuan AI untuk mempercepat kurasi metadata koleksi.

II. METODE

Penelitian ini menggunakan metode pengembangan perangkat lunak Agile dengan pendekatan Scrum. Metode ini dipilih karena kebutuhan sistem bersifat bertahap dan memerlukan evaluasi berulang terhadap alur pengguna, akademisi, dan admin.



Gambar 1. Tahapan Metode Agile Scrum (Sumber: diadaptasi dari Alam dan Azizah [16]

Tahapan penelitian ditunjukkan pada Gambar 1. Gambar tersebut disesuaikan dari pendekatan Scrum yang menempatkan product backlog, sprint backlog, sprint, dan implementation sebagai alur utama pengembangan sistem informasi [16]. Pada penelitian ini, tahapan tersebut diperluas menjadi identifikasi masalah, pengumpulan data, product backlog, sprint planning, sprint backlog, sprint development, finished work, sprint review dan retrospective, serta pengujian sistem agar selaras dengan kebutuhan sistem reservasi dan distribusi koleksi pustaka digital.

1. **Identifikasi masalah:**

   Tahap ini digunakan untuk merumuskan kesenjangan utama, yaitu belum terpusatnya distribusi karya akademik, belum optimalnya fitur reservasi atau bookmarking koleksi, kebutuhan validasi kontributor, serta kebutuhan pengendalian akses pada koleksi berbayar. Hasil identifikasi diterjemahkan menjadi kebutuhan katalog, detail koleksi, reservasi, transaksi manual, verifikasi akademisi, verifikasi koleksi, integrasi AI, rujukan sumber eksternal, dan payout.

2. **Pengumpulan data:**
   Data dikumpulkan melalui studi literatur dan observasi kebutuhan sistem. Studi literatur digunakan untuk memahami perpustakaan digital, arsitektur API-driven, Firebase Firestore, Scrum, Black Box Testing, dan UAT. Observasi dilakukan terhadap alur pengguna dalam mencari koleksi, menyimpan koleksi, mengunduh dokumen, mengajukan verifikasi akademisi, mengunggah karya, serta memproses transaksi berbayar.
3. **Product backlog:**
   Kebutuhan hasil identifikasi disusun menjadi daftar fitur yang diprioritaskan. Backlog meliputi autentikasi, role, katalog, reservasi, riwayat unduhan, verifikasi akademisi, unggah koleksi, verifikasi koleksi, rujukan sumber eksternal, bantuan AI, transaksi manual, rekening platform, dan payout.
4. **Sprint planning:**
   Backlog diprioritaskan dalam sprint agar fitur dasar seperti autentikasi, role, katalog, dan reservasi dikerjakan lebih awal sebelum fitur yang bergantung pada data tersebut, seperti transaksi, verifikasi koleksi, integrasi AI, rujukan sumber eksternal, dan payout.
5. **Sprint backlog:**
   Sprint backlog berisi fitur yang dipilih untuk dikerjakan pada setiap sprint. Susunan ini membantu pengembangan tetap fokus pada target sprint dan memastikan dependensi antarmodul tidak saling bertabrakan.
6. **Sprint development:**
   Pada tahap development, fitur diimplementasikan pada frontend Nuxt 3 dan backend Express.js melalui endpoint REST API. Struktur Firestore dan integrasi Google Drive API juga dikembangkan sesuai kebutuhan setiap modul.
7. **Finished work:**
   Setiap sprint menghasilkan modul yang dapat diuji, seperti autentikasi, katalog, reservasi, pengajuan akademisi, transaksi, dashboard, serta payout. Hasil sprint kemudian digunakan sebagai dasar evaluasi pada tahap berikutnya.
8. **Sprint review dan retrospective:**
   Setiap hasil sprint dievaluasi untuk memastikan alur antarmuka, validasi backend, struktur Firestore, dan hak akses role berjalan konsisten. Retrospective digunakan untuk memperbaiki prioritas, validasi, dan pengalaman pengguna pada sprint lanjutan.
9. **Implementasi dan pengujian sistem**:
   Implementasi akhir diuji menggunakan Black Box Testing untuk memeriksa kesesuaian input-output fitur, serta User Acceptance Testing untuk menilai penerimaan pengguna terhadap tampilan, navigasi, katalog, reservasi, transaksi, dashboard, dan kelayakan sistem.
   Pengujian dilakukan menggunakan Black Box Testing dan User Acceptance Testing. Black Box Testing digunakan untuk menguji fungsi utama berdasarkan skenario input dan output, mulai dari registrasi, login, pencarian koleksi, reservasi, unggah koleksi, transaksi, verifikasi admin, sampai payout. User Acceptance Testing digunakan untuk mengetahui tingkat penerimaan pengguna. UAT melibatkan 20 responden yang terdiri dari 15 mahasiswa, 3 dosen atau akademisi, dan 2 responden umum dengan skala Likert 1 sampai 4. Penggunaan UAT mengacu pada penelitian pengujian penerimaan sistem informasi dan aplikasi digital yang menilai kelayakan sistem dari perspektif pengguna akhir [13] [20][23].

III. HASIL DAN PEMBAHASAN

A. Hasil identifikasi kebutuhan dan product backlog

Hasil identifikasi menunjukkan bahwa sistem perlu mendukung tiga aktor utama, yaitu pengguna umum, akademisi, dan admin. Pengguna membutuhkan akses katalog, detail koleksi, reservasi atau bookmark, unduhan, pembelian berbayar, dan riwayat transaksi. Akademisi membutuhkan verifikasi akun, unggah karya, kelola karya, rekening, pemantauan transaksi penjualan, dan payout. Admin membutuhkan pengelolaan pengguna, kategori, koleksi, transaksi, verifikasi akademisi, rekening platform, rujukan sumber eksternal, dan payout. Hasil pengumpulan data melalui studi literatur dan observasi kemudian digunakan sebagai dasar penyusunan product backlog.

Tabel 1. Kebutuhan sistem berdasarkan aktor

Aktor	Hasil kebutuhan
Pengguna	Katalog, detail koleksi, bookmark, unduhan, pembelian berbayar, dan riwayat transaksi.
Akademisi	Verifikasi akun, unggah koleksi, kelola karya, rekening, transaksi penjualan, dan payout.
Admin	Kelola pengguna, kategori, koleksi, transaksi, verifikasi akademisi, rekening platform, dan payout.

B. Product Backlog, Sprint Planning, dan Sprint Backlog

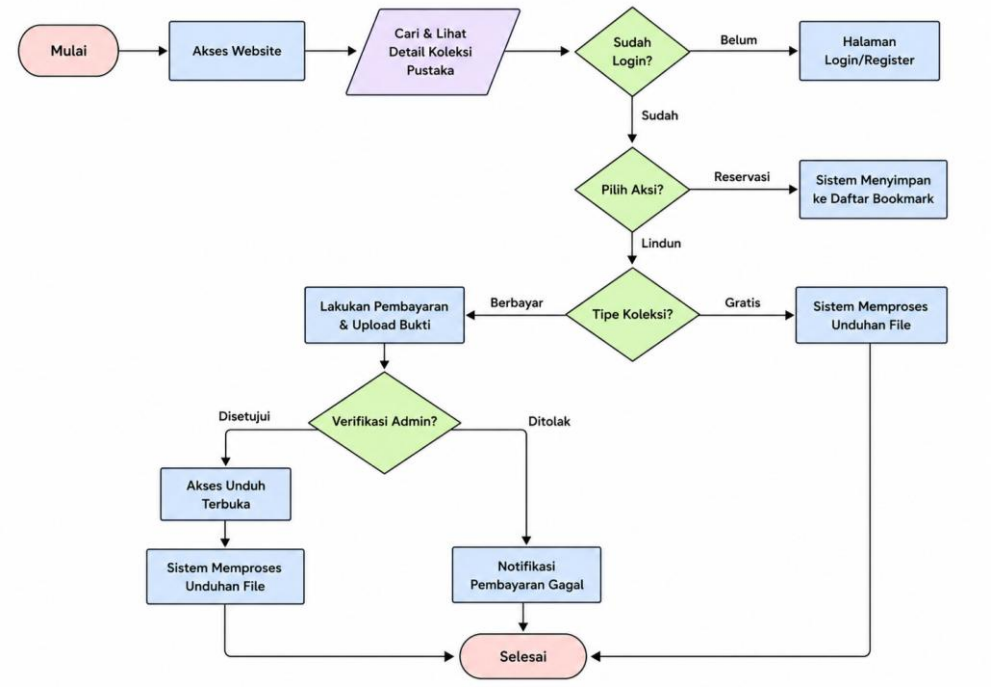
Kebutuhan pada Tabel 1 dipetakan menjadi product backlog agar pengembangan mengikuti tahapan Scrum. Product backlog dikelompokkan ke dalam sprint yang saling bergantung. Autentikasi dan role dikerjakan lebih awal karena seluruh fitur lanjutan memerlukan validasi pengguna. Katalog dan reservasi dikerjakan sebelum transaksi karena pembelian dan unduhan membutuhkan data koleksi yang sudah published. Verifikasi akademisi dan verifikasi koleksi ditempatkan sebelum payout karena pendapatan hanya dapat dihitung dari koleksi akademisi yang sah dan transaksi yang telah dikonfirmasi.

Tabel 2. Sprint backlog pengembangan sistem

Sprint	Fokus backlog	Output sprint
1	Autentikasi dan manajemen role	Registrasi, login, JWT, bcrypt, middleware role
2	Katalog, kategori, dan eksplorasi koleksi	Halaman katalog, pencarian, filter, detail koleksi.
3	Reservasi, bookmark, dan riwayat unduhan	Penyimpanan koleksi, status reservasi, pencatatan unduhan
4	Pengajuan akademisi dan unggah karya	Pengajuan akademisi dan unggah karya
5	Verifikasi koleksi, rujukan eksternal, dan bantuan AI	Kurasi koleksi, metadata awal, dan mode rujukan sumber asli
6	Transaksi manual dan validasi pembayaran	Upload bukti pembayaran dan validasi status transaksi
7	Rekening platform, pendapatan, dan payout	Pengelolaan rekening, saldo akademisi, dan permintaan payout

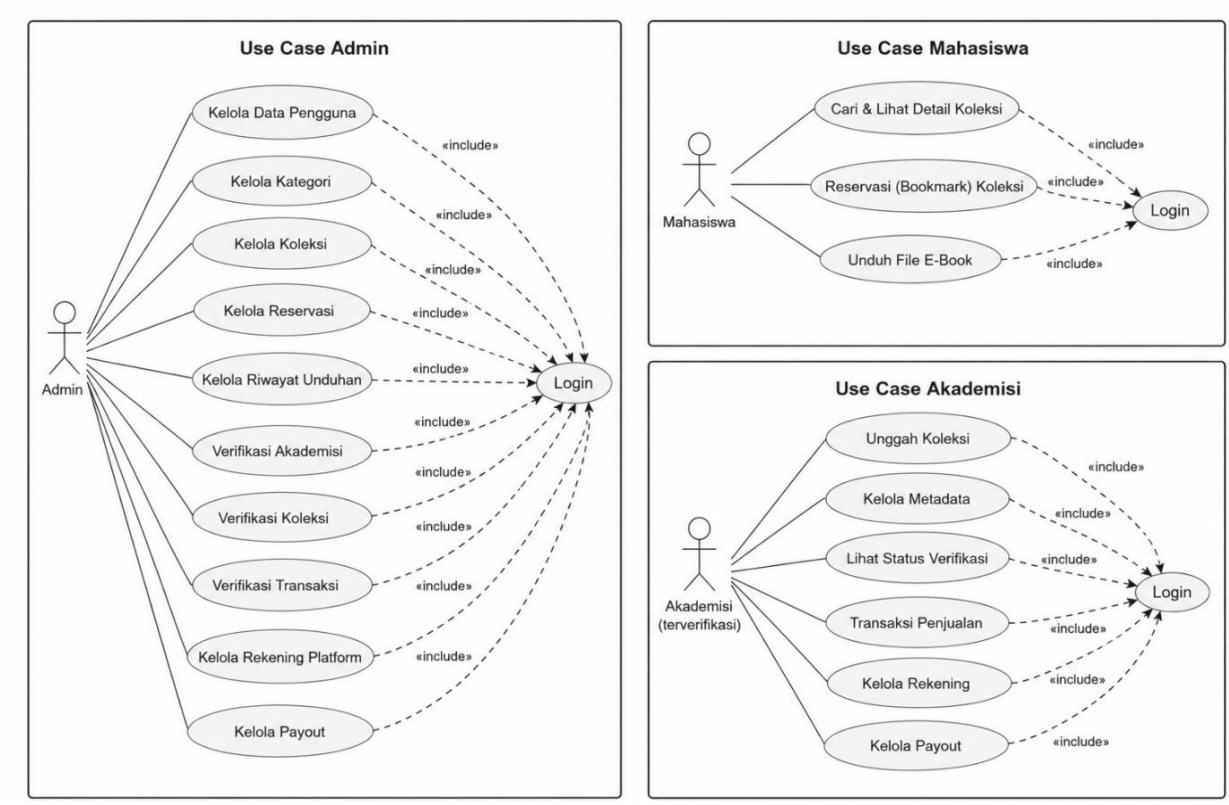
C. Sprint Development dan Finished Work

Pada tahap sprint development, backlog diterjemahkan menjadi rancangan alur, arsitektur, basis data, endpoint backend, dan antarmuka frontend. Hasil sprint yang telah selesai ditampilkan melalui flowchart, use case, arsitektur API-driven, class diagram Firestore, sequence transaksi, dan implementasi antarmuka. Urutan penyajian ini mengikuti alur pada Gambar 1 sehingga hasil dan pembahasan tidak terlepas dari metode Scrum yang digunakan.



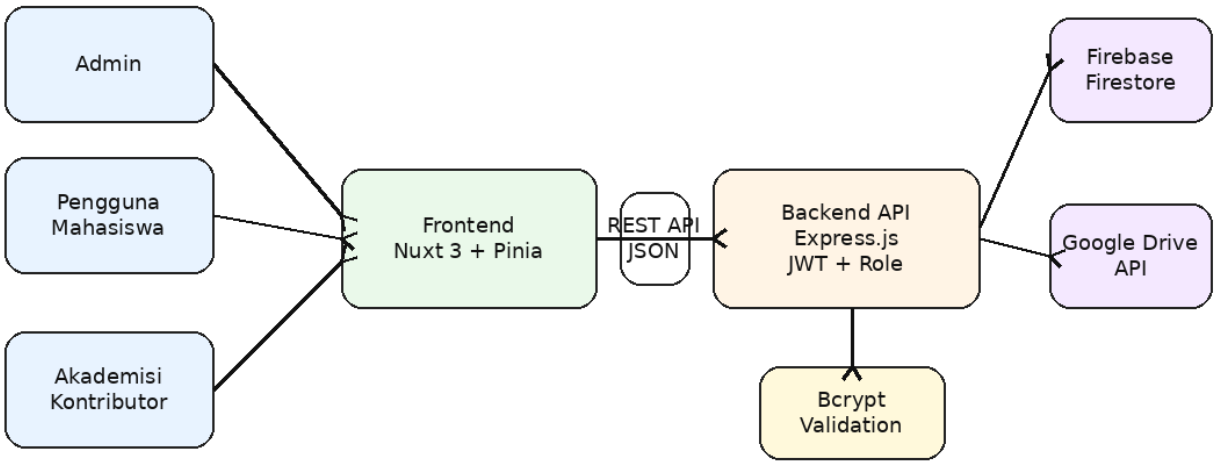
Gambar 2. Flowchart alur pengguna dalam sistem reservasi koleksi

Gambar 2 menunjukkan alur pengguna ketika mengakses katalog, melihat detail koleksi, melakukan reservasi, dan memperoleh akses unduhan. Untuk koleksi gratis, pengguna dapat mengunduh setelah sistem memvalidasi akun dan kelengkapan profil. Untuk koleksi berbayar, pengguna diarahkan pada pembayaran manual dan unggah bukti pembayaran. Akses unduhan baru dibuka setelah admin mengonfirmasi transaksi. Alur admin dan akademisi mengikuti pola yang sama: admin bertindak sebagai validator, sedangkan akademisi bertindak sebagai kontributor yang karya dan rekeningnya tetap melalui kontrol sistem.



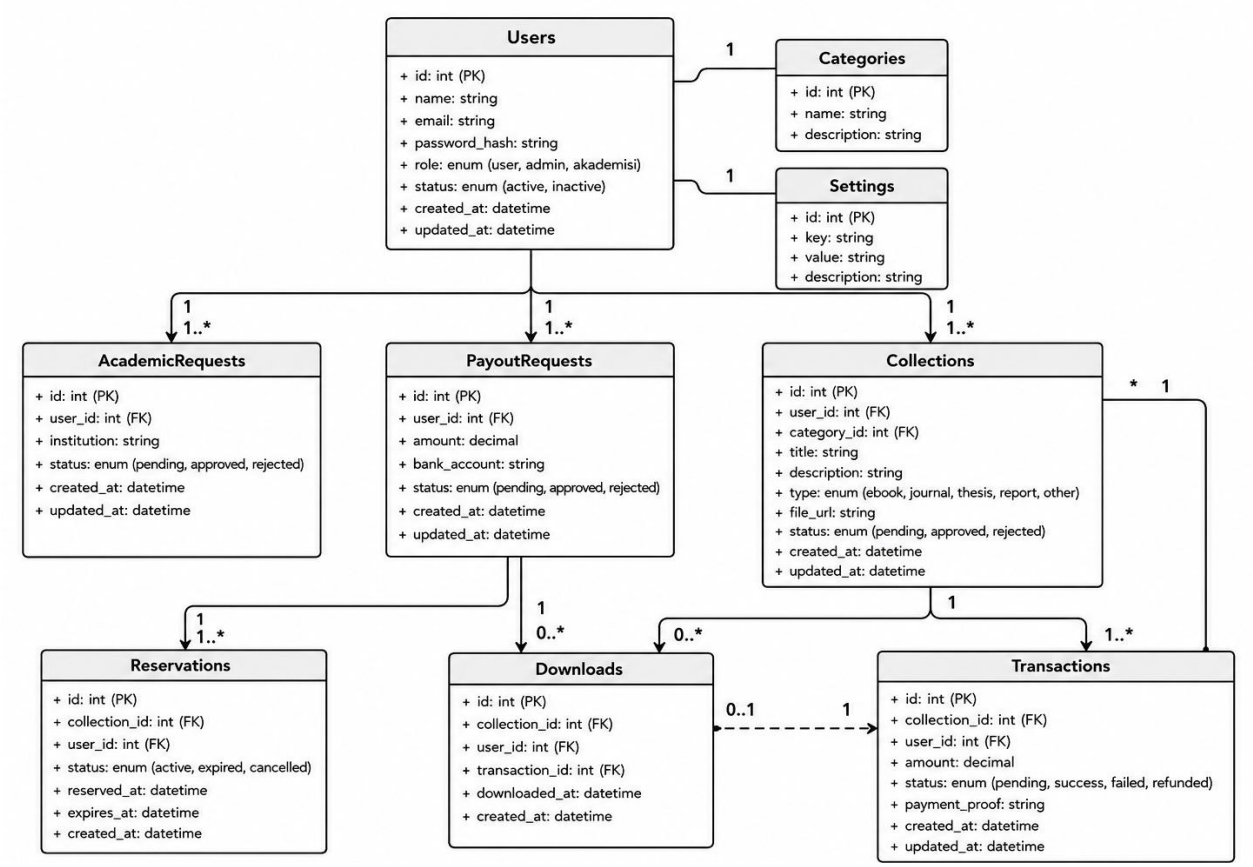
Gambar 3. Use Case Diagram Sistem Berdasarkan Aktor

Use case pada Gambar 3 memperlihatkan bahwa sistem dirancang sebagai workflow lintas aktor. Pengguna umum dapat mengakses katalog, menyimpan koleksi, mengunduh koleksi gratis, membeli koleksi berbayar, dan mengajukan verifikasi akademisi. Akademisi dapat mengunggah karya, memantau status publikasi, melihat transaksi penjualan, serta mengajukan payout. Admin mengelola validasi utama, yaitu akun akademisi, kelayakan koleksi, transaksi pembayaran manual, rekening platform, dan payout.



Gambar 4. Arsitektur API-driven sistem reservasi dan distribusi koleksi pustaka digital

Arsitektur API-driven pada Gambar 4 memisahkan antarmuka Nuxt 3 dari backend Express.js. Seluruh permintaan data dikirim melalui REST API, kemudian backend memvalidasi token JWT, role pengguna, payload, dan status data sebelum mengakses Firebase Firestore atau Google Drive API. Pemisahan ini membuat frontend tidak berkomunikasi langsung dengan basis data sehingga logika autentikasi, otorisasi, transaksi, dan verifikasi dapat dikendalikan secara konsisten pada backend.

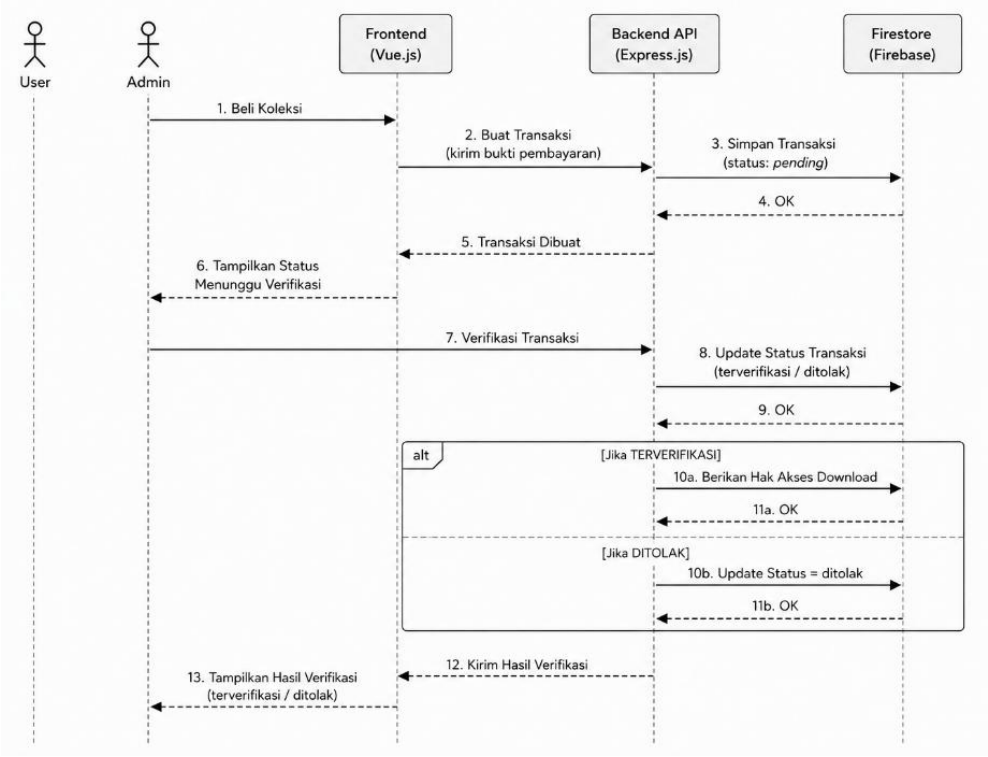


Gambar 5. Class Diagram Struktur Collection Firestore

Struktur data Firestore pada Gambar 5 terdiri atas collection users, collections, reservations, transactions, academic_requests, settings, payoutRequests, downloads, dan collection pendukung lain. Relasi antardata dibangun melalui referensi ID seperti userId, collectionId, buyerId, sellerId, ownerId, academicId, dan payoutRequestId. Struktur ini sesuai dengan implementasi kode sumber karena controller backend memproses data berdasarkan ID tersebut sebelum mengembalikan respons JSON ke frontend.

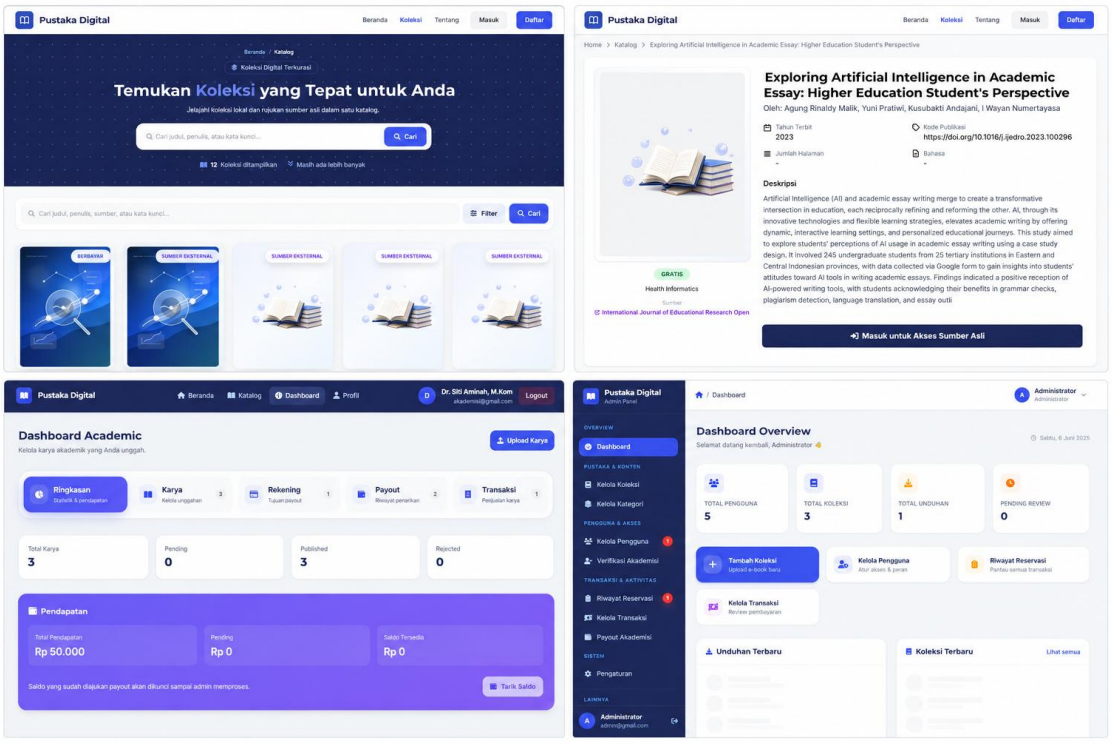
Implementasi sistem dilakukan pada dua aplikasi utama. Backend menggunakan Express.js, Firebase Admin, Google APIs, JWT, bcrypt, multer, Joi, helmet, CORS, dan rate limiter. Frontend menggunakan Nuxt 3, Vue 3, Pinia, Tailwind CSS, dan middleware autentikasi. Route backend meliputi `/api/auth`, `/api/users`, `/api/collections`, `/api/reservations`, `/api/downloads`, `/api/transactions`, `/api/academic-requests`, `/api/payouts`, `/api/admin`, `/api/academic/dashboard`, dan endpoint katalog eksternal. Susunan ini menunjukkan bahwa fitur pada product backlog telah diterjemahkan menjadi modul API yang terpisah namun saling terhubung.

Pada modul reservasi, sistem menyimpan data bookmark dengan status active atau cancelled sehingga riwayat aktivitas tetap dapat dipertahankan. Untuk dashboard akademisi, endpoint `/api/reservations/academic/mine` mengambil koleksi berdasarkan ownerId pengguna akademisi, memecah daftar collectionId agar sesuai dengan batas query Firestore, memfilter reservasi aktif, lalu menggabungkan data pengguna dan koleksi. Implementasi ini membuat akademisi dapat melihat karya mana yang disimpan oleh pengguna tanpa perlu membaca seluruh data reservasi secara tidak efisien.



Gambar 6. Sequence Diagram Transaksi Koleksi Berbayar

Gambar 6 menunjukkan alur transaksi koleksi berbayar. Pengguna memilih koleksi, sistem menampilkan informasi rekening platform, pengguna mengunggah bukti pembayaran, backend menyimpan transaksi dengan status pending, dan admin melakukan verifikasi. Jika bukti pembayaran valid, status berubah menjadi confirmed dan akses unduhan dibuka. Jika tidak valid, status menjadi rejected. Pendekatan ini sesuai dengan batasan penelitian karena sistem belum menggunakan payment gateway otomatis.



Gambar 7. Implementasi Antarmuka Pengguna

Implementasi antarmuka pada Gambar 7 memperlihatkan halaman publik, dashboard pengguna, pengajuan verifikasi akademisi, dan dashboard akademisi. Halaman publik digunakan untuk mengenalkan platform dan katalog. Dashboard pengguna mengintegrasikan reservasi, riwayat unduhan, pembelian, dan aktivitas. Dashboard akademisi menyediakan unggah karya, status publikasi, rekening, transaksi, reservasi karya, dan payout. Dashboard admin digunakan untuk mengelola pengguna, kategori, koleksi, transaksi, verifikasi, rekening platform, dan payout.

D. Integrasi API, Rujukan Sumber Eksternal, dan Fitur AI

Pendekatan API-driven memungkinkan platform lain berintegrasi melalui pola request-response berbasis HTTP dan JSON. Pada integrasi sederhana, platform lain dapat membaca data katalog publik melalui endpoint koleksi atau pencarian eksternal. Pada integrasi yang membutuhkan aksi pengguna, seperti reservasi, transaksi, atau akses sumber, sistem memerlukan autentikasi JWT agar identitas pengguna, role, dan kelengkapan profil dapat divalidasi. Dengan pola ini, backend berfungsi sebagai gateway yang menjaga aturan bisnis, sementara frontend atau platform lain hanya mengonsumsi respons API yang sudah tervalidasi.

Sistem juga menyediakan mode rujukan sumber eksternal. Jika karya sudah tersedia pada repository, jurnal, publisher, atau platform resmi lain, sistem tidak perlu mengunggah ulang file lokal. Backend menyimpan metadata dan tautan sumber asli, sedangkan pengguna diarahkan ke platform asal setelah aktivitas akses dicatat. Mekanisme ini membantu memperluas katalog tanpa mengambil alih hak distribusi dari sumber asli. Untuk pengembangan lebih lanjut, integrasi platform eksternal dapat diperkuat dengan dokumentasi API publik, API key, pembatasan kuota, dan format kontrak data yang stabil.

Fitur AI diterapkan pada modul unggah koleksi untuk membantu kurasi awal. Endpoint `/api/collections/ai/analyze` menerima PDF dari admin atau akademisi, kemudian menghasilkan metadata awal seperti judul, penulis, tahun, bahasa, jumlah halaman, deskripsi, kategori yang disarankan, kata kunci, dan prompt cover. Modul AI juga melakukan publication review, yaitu mendeteksi indikasi bahwa karya sudah pernah dipublikasikan melalui DOI, ISBN, ISSN, nama jurnal, nama penerbit, repository, atau platform seperti Crossref, GARUDA, SINTA, ResearchGate, Zenodo, Google Scholar, IEEE, ACM, Springer, Elsevier, dan Open Journal Systems. Jika indikasi publikasi ditemukan, sistem mengarahkan pengunggah untuk menggunakan mode rujukan sumber asli. Endpoint /api/collections/ai/cover digunakan untuk membuat cover koleksi berbasis metadata. Dengan demikian, AI tidak menggantikan validasi admin, tetapi mempercepat pengisian metadata dan membantu menjaga kepatuhan distribusi koleksi.

E. Sprint Review, Retrospective, dan Hasil Pengujian

Sprint review dilakukan untuk memastikan setiap finished work telah sesuai dengan target sprint. Retrospective digunakan untuk memperbaiki alur fitur yang masih membutuhkan penguatan, terutama pada validasi role, pembatasan akses koleksi berbayar, serta status transaksi dan payout. Setelah seluruh modul terintegrasi, pengujian dilakukan menggunakan Black Box Testing dan User Acceptance Testing.

Tabel 3. Ringkasan hasil pengujian Black Box

Kelompok	Total	Valid	Persentase
Auth dan profil	4	4	100%
Katalog dan reservasi	4	4	100%
Unggah koleksi	3	3	100%
Verifikasi akademisi	2	2	100%
Transaksi berbayar	5	5	100%
Payout dan rekening	2	2	100%
Total	20	20	100%

Persentase keberhasilan pengujian dihitung dengan rumus Persentase = (skenario valid / total skenario) x 100%. Berdasarkan pengujian terhadap 20 skenario, seluruh skenario memperoleh hasil valid sehingga persentase keberhasilan mencapai 100%. Temuan penting dari pengujian adalah sistem mampu menolak akses unduhan koleksi berbayar saat transaksi masih pending dan membuka akses setelah transaksi confirmed.

User Acceptance Testing dilakukan untuk mengetahui penerimaan pengguna terhadap sistem. Responden diminta mencoba sistem informasi reservasi koleksi pustaka digital melalui website yang telah disediakan, kemudian mengisi kuesioner dengan skala Likert 1 sampai 4. Aspek yang dinilai meliputi kemudahan penggunaan, tampilan antarmuka, katalog, pencarian, reservasi, unduhan, transaksi manual, status transaksi, responsivitas, dashboard, kesesuaian kebutuhan, dan kelayakan sistem. Penggunaan UAT pada penelitian ini diperkuat oleh penelitian sebelumnya yang menggunakan UAT untuk menilai penerimaan sistem dari perspektif pengguna akhir [13][21][23].

Tabel 4. Hasil rekapitulasi User Acceptance Testing

No	Aspek Penilaian	Total Skor	Skor Maksimum	Persentase
1	Tampilan sistem mudah dipahami	80	80	100,00%
2	Navigasi menu mudah digunakan	79	80	98,75%
3	Informasi katalog koleksi jelas	78	80	97,50%
4	Fitur pencarian membantu pengguna	79	80	98,75%
5	Fitur reservasi/bookmark mudah digunakan	79	80	98,75%
6	Proses unduhan koleksi mudah dipahami	80	80	100,00%
7	Alur transaksi manual mudah dipahami	79	80	98,75%
8	Informasi status transaksi jelas	79	80	98,75%
9	Sistem berjalan responsif	80	80	100,00%
10	Dashboard pengguna mudah dipahami	79	80	98,75%
11	Sistem sesuai kebutuhan pengguna	78	80	97,50%
12	Sistem layak digunakan	78	80	97,50%
	Total	948	960	98,75%

Skor maksimum diperoleh dari jumlah responden dikalikan jumlah aspek penilaian dan skor tertinggi pada skala Likert, yaitu 20 x 12 x 4 = 960. Persentase UAT dihitung dengan rumus (total skor / skor maksimum) x 100%, sehingga diperoleh (948 / 960) x 100% = 98,75%. Nilai ini berada pada interval 76% sampai 100% dan termasuk kategori sangat layak. Hasil tersebut menunjukkan bahwa sistem dapat diterima oleh pengguna sebagai media reservasi dan distribusi koleksi pustaka digital berbasis web.

F. Pembahasan

Hasil implementasi menunjukkan bahwa pendekatan API-driven sesuai untuk sistem dengan alur multi-role. Frontend berfokus pada pengalaman pengguna, sedangkan backend menjadi pusat validasi dan pengelolaan logika bisnis. Pemisahan ini membuat autentikasi, otorisasi, transaksi, verifikasi, integrasi sumber eksternal, dan fitur AI dapat dikelola secara konsisten melalui API. Sistem juga mendukung distribusi hybrid. Koleksi gratis dapat diakses lebih sederhana, koleksi berbayar dikendalikan melalui verifikasi transaksi manual, dan koleksi yang sudah tersedia pada platform lain dapat dicatat sebagai rujukan sumber asli.

Tabel 5. Kesesuaian kebutuhan dan implementasi

Kebutuhan	Implementasi	Status
Reservasi koleksi	Endpoint reservations dan store bookmark	Terpenuhi
Distribusi hybrid	Tipe koleksi gratis dan berbayar	Terpenuhi
Validasi kontributor	academic_requests dan perubahan role	Terpenuhi
Verifikasi transaksi	status pending, confirmed, dan rejected	Terpenuhi
Payout akademisi	payoutRequests dan validasi admin	Terpenuhi

Dibandingkan sistem perpustakaan digital yang hanya menyediakan pencarian dan unduhan, sistem ini menambahkan reservasi personal, publikasi karya akademik, rujukan sumber eksternal, transaksi manual, pencatatan aktivitas unduhan, bantuan AI untuk kurasi awal, serta pencairan pendapatan akademisi. Keterbatasan yang masih ada adalah pembayaran belum otomatis dan keamanan akses file berbayar masih dapat ditingkatkan melalui secure download atau token unduhan sementara. Selain itu, integrasi API untuk platform lain masih perlu diperkuat dengan dokumentasi endpoint, API key, manajemen kuota, dan audit log agar dapat digunakan secara lebih luas.

IV. SIMPULAN DAN SARAN

A. Simpulan

Sistem informasi reservasi dan distribusi koleksi pustaka digital berbasis web berhasil dikembangkan menggunakan pendekatan API-driven dengan Nuxt 3/Vue 3, Express.js, Firebase Firestore, dan Google Drive API. Sistem mampu mengelola katalog koleksi, reservasi atau bookmarking, unggah koleksi akademisi, verifikasi akademisi, verifikasi koleksi, transaksi pembayaran manual, riwayat unduhan, rujukan sumber eksternal, pengaturan rekening platform, payout akademisi, serta bantuan AI untuk analisis metadata PDF, deteksi indikasi sumber publikasi, dan pembuatan cover koleksi. Pengujian Black Box pada 20 skenario menunjukkan bahwa seluruh fitur utama berjalan sesuai hasil yang diharapkan, sehingga sistem dinilai telah memenuhi kebutuhan fungsional utama. Hasil User Acceptance Testing terhadap 20 responden memperoleh persentase penerimaan sebesar 98,75% dan termasuk kategori sangat layak. Dengan demikian, sistem dapat diterima oleh pengguna sebagai platform reservasi serta distribusi koleksi pustaka digital yang mendukung workflow pengguna, akademisi, dan admin secara terpadu.

B. Saran

Pengembangan selanjutnya disarankan untuk menambahkan integrasi payment gateway agar transaksi koleksi berbayar dapat diverifikasi otomatis, menerapkan secure download atau token unduhan sementara untuk meningkatkan keamanan akses file, menambahkan notifikasi realtime untuk transaksi dan verifikasi, menyediakan dokumentasi API publik beserta API key agar integrasi dengan platform lain lebih terkontrol, serta menambahkan audit log untuk mencatat aktivitas penting admin, akademisi, dan pengguna. Fitur AI juga dapat dikembangkan dengan validasi metadata yang lebih transparan agar hasil analisis tetap mudah diaudit oleh admin.

UCAPAN TERIMA KASIH

Terima kasih disampaikan kepada Universitas Muhammadiyah Sidoarjo, Program Studi Informatika, serta seluruh responden yang telah membantu proses pengujian sistem dan pengisian kuesioner User Acceptance Testing.

REFERENSI

[1] J. Santi, I. P. Windasari, and A. B. Prasetijo, “Perancangan Sistem Informasi Perpustakaan Berbasis Web dalam Upaya Mewujudkan Digitalisasi Sistem Administrasi Perpustakaan di SD Negeri 1 Tambak,” Jurnal Teknik Komputer, vol. 1, no. 4, pp. 192-198, 2023, doi: 10.14710/jtk.v1i4.37545.

[2] Setyarini, P. Kalisa, and A. D. Cahyaningtyas, “Manajemen e-book di UPT Perpustakaan Universitas Negeri Semarang melalui sistem aplikasi,” Jurnal Ilmu Perpustakaan dan Informasi, vol. 10, no. 2, pp. 41-57, 2021.

[3] R. H. Saputra et al., “Rancang Bangun Perpustakaan Buku Digital (E-Book) Berbasis Web,” Jurnal El-Pustaka, vol. 2, no. 2, pp. 58-70, 2021, doi: 10.24042/el-pustaka.v2i2.10175.

[4] D. Zulfinar, “Rancang Bangun Sistem Informasi Pustaka Online Berbasis Web untuk Kampus STMIK Indonesia Banda Aceh,” vol. 3, no. 1, pp. 36-48, 2023.

[5] S. Wahyuntini, “Analisis Sistem Katalogisasi E-Book di UPT Perpustakaan ISI Yogyakarta,” 2024.

[6] P. Subekti and A. Pratama, “Analisis dan Perancangan Sistem Informasi Perpustakaan Digital Berbasis Web,” Data Science and Information System (DIMIS), vol. 2, no. 2, pp. 70-79, 2024, doi: 10.58602/dimis.v2i2.123.

[7] Universitas Muhammadiyah Sidoarjo, “Perpustakaan UMSIDA Hadirkan Inovasi E-Library di Aplikasi MyUmsida,” 2023.

[8] M. A. Abdurrauf, “Pengembangan frontend sistem informasi manajemen rekrutmen dan komunitas menggunakan framework Nuxt,” 2024.

[9] R. Namruddin, M. Zain, and Y. R. D., “Sistem Manajemen Perpustakaan Berbasis Web Menggunakan Framework Laravel dan Vue.js melalui Komunikasi API,” vol. 8, no. 12, pp. 29-36, 2024.

[10] L. Ramadhani, R. Amalia, and F. Puspita, “Implementasi Firebase Realtime Database Pada Aplikasi Integrated Perpustakaan SMK Prestasi Prima,” Seminar Nasional Riset dan Teknologi (SEMNAS RISTEK), pp. 283-288, 2021.

[11] R. Saputra et al., “Implementasi Metode Agile Dalam Pengembangan Sistem Informasi Perpustakaan Berbasis Web Pada SMA Negeri 1 Sigli,” Jurnal Literasi Informatika, vol. 3, no. 3, 2024.

[12] M. P. A. Ginting, “Pengujian Aplikasi Berbasis Web Data Ska Menggunakan Metode Black Box Testing,” vol. 2, no. 1, pp. 41-48, 2024.

[13] H. Muhammad et al., “Pengguna (User Acceptance Testing) Pada Sistem Informasi Akademik EMACCA Universitas Teknologi AKBA Makassar,” vol. 3, no. 2, pp. 84-91, 2025.

[14] A. Izulhaq, U. Indahyanti, and I. R. I. Astutik, “Sistem Informasi Pemesanan Produk Percetakan Berbasis Web Menggunakan Metode Waterfall,” KLIK: Kajian Ilmiah Informatika dan Komputer, vol. 4, no. 1, pp. 486-496, 2023, doi: 10.30865/klik.v4i1.1146.

[15] R. R. Sandy, A. Eviyanti, A. W. Azinar, and H. Setiawan, “Sistem Informasi Manajemen Aset Berbasis Web di Perusahaan Manufaktur Alat Kesehatan,” JATI (Jurnal Mahasiswa Teknik Informatika), vol. 10, no. 2, pp. 1891-1899, 2026, doi: 10.36040/jati.v10i2.17225.

[16] M. N. B. Alam and N. L. Azizah, “Perancangan Sistem Informasi Desa Menggunakan Metode Scrum,” Indonesian Journal of Applied Technology, vol. 1, no. 3, 2024, doi: 10.47134/ijat.v1i3.3106.

[17] S. A. Fadillah, N. Chandra, and C. Rivatunisa, “Implementasi Agile Scrum Pada Pembuatan Website Sistem Informasi Manajemen Kuliner,” Decode: Jurnal Pendidikan Teknologi Informasi, vol. 4, no. 1, pp. 301-315, 2024, doi: 10.51454/decode.v4i1.357.

[18] I. Tahyudin and Z. I. Sholihati, “Pengembangan Aplikasi Tiga-Tingkat menggunakan Metode Scrum pada Aplikasi Presensi Karyawan Glints Academy,” Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), vol. 6, no. 1, pp. 169-176, 2022, doi: 10.29207/resti.v6i1.3793.

[19] C. F. Alqodri, “Pengembangan Frontend Website pada Platform Survey Online menggunakan Agile Scrum,” Prosiding Seminar Implementasi Teknologi Informasi dan Komunikasi, vol. 3, no. 2, 2024, doi: 10.31284/p.semtik.2024-2.6239.

[20] I. Afrianto, A. Heryandi, A. Finandhita, and S. Atin, “User Acceptance Test For Digital Signature Application In Academic Domain To Support The Covid-19 Work From Home Program,” IJISTECH (International Journal of Information System and Technology), vol. 5, no. 3, pp. 270-280, 2021, doi: 10.30645/ijistech.v5i3.132.

[21] W. Wulandari, N. Nofiyani, and H. Hasugian, “User Acceptance Testing (UAT) Pada Electronic Data Preprocessing Guna Mengetahui Kualitas Sistem,” Jurnal Mahasiswa Ilmu Komputer, vol. 4, no. 1, pp. 20-27, 2023, doi: 10.24127/ilmukomputer.v4i1.3383.

[22] I. P. Soko, “The Evaluation of web-based and android face-to-face tutorial application using User Acceptance Testing,” Journal of World Science, 2022.

[23] A. K. Santoso, “Analysis and Design of ERP Information Systems Using User Acceptance Test Assessment,” Bit-Tech, 2025.

Conflict of Interest Statement:
The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.