Vol. 11, No. 2, Oktober 2026, Page. 621-643
https://journal.uin-alauddin.ac.id/index.php/instek
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

Design and Evaluation of Enterprise Architecture for Academic Information System and Repository Integration Using TOGAF ADM

Fauziah[1], Yulia[2], Wilda Putri Sabila[3], Mahara Bengi[4], Rahman Pradipta[5], Nabila Sasgita[6]

[1-2]Sistem Informasi, Institut Teknologi Rokan Hilir, Indonesia
[3]Sistem Informasi, Institut Teknologi dan Bisnis Muhammadiyah Sarolangun, Indonesia
[4-5]Informatika, Universitas Samudra, Indonesia
[6]Teknologi Informasi, Universitas Merangin, Indonesia

Email: [1]fauziahnurgayo@gmail.com, [2]mrsyulia98@gmail.com, [3]wildasabilaa@gmail.com, [4]maharabengi@unsam.ac.id, [6]rahman.pradipta@unsam.ac.id, [7]nabilasasgita@gmail.com

Abstract

This study designs and evaluates a TOGAF ADM-based enterprise architecture for integrating an Academic Information System and an academic repository while preserving the functional autonomy of both applications. The architecture positions SIAKAD as the authoritative source for selected academic master data and exposes approved attributes through the confirmed public HTTPS REST API endpoint https://api.siakad.ac.id/v1. The evaluation combines stakeholder confirmation by five operational representatives 4.74/5.00, assessment by two external EA experts 4.67/5.00, functional and security tests, synchronization records for 20 students, a 25-sample JMeter evidence set, and 25 paired operational observations performed by five operators. Technical tests recorded 20/20 successful valid requests, four HTTP 404 responses for unregistered identifiers, one HTTP 400 response for malformed input, HTTP 401 for missing/invalid tokens, and an HTTP 403 response for insufficient role privileges. The synchronization workbook records 160 attribute-level MATCH markers across eight fields and 20 records; the SQL snapshot lists 20 post-sync records but its displayed after-sync query omits two mapped fields. The JMeter dashboard reports 183.12 ms mean response time (170–194 ms), whereas recalculation from the separate raw-log screenshot gives 184.36 ms (174–196 ms); the artifacts cannot yet be confirmed as the same execution. Mean operational time decreased from 94.60 s to 11.408 s (87.94%; $t(24)=74.05$; $p<0.001$). Because the same five operators performed As-Is before To-Be, the fixed sequence and possible learning effect limit causal interpretation. The improvement is specific to the observed workflow and test conditions, not a universal estimate for other institutions.

Keywords: Enterprise Architecture, TOGAF ADM, Academic Information System, System Integration, REST API, Academic Repository.

This work is an open access article and licensed under a Creative Commons Attribution-NonCommercial ShareAlike 4.0 International (CC BY-NC-SA 4.0)

1. INTRODUCTION

Higher-education institutions increasingly depend on digital platforms to support academic administration, learning, research, library services, and institutional management. Digital transformation can improve service reach and decision support, but it also exposes problems created by fragmented applications, institutional silos, inconsistent data practices, and weak coordination across autonomous units [1], [2]. Enterprise Architecture (EA) addresses these conditions by aligning business processes, information, applications, and technology within a coordinated transformation view; higher-education reviews and implementation studies consequently position EA as a useful mechanism for planning integrated digital services rather than developing isolated applications [3], [4].

Preliminary mapping in the present case identifies eight potentially overlapping attributes: student identification number, student name, study program, faculty, final-project title, supervisor, graduation year, and graduation status. The separation of systems may lead to repeated input, nomenclature differences, asynchronous updates, and inconsistent metadata. Higher-education data-governance studies similarly emphasize the importance of accurate, current, synchronized academic data and clearly assigned responsibilities for institutional data management [5], [2]. Institutional-repository

research further shows that repository platforms have their own software, metadata, maintenance, and service requirements [6]. These conditions indicate that the problem is not merely an application-interface issue but an architectural problem involving data ownership, application interaction, repository autonomy, and technology integration.

TOGAF ADM has been widely applied to develop enterprise-architecture blueprints because it structures the transition from architecture principles and vision to business, information-systems, technology, and implementation-oriented decisions [7], [8]. Recent education and higher-education studies have used TOGAF to design student-affairs architecture, integrated university management services, cross-domain information-system integration, and digital-transformation blueprints [9], [10]. These studies support the suitability of a phase-based EA approach for diagnosing As-Is fragmentation and defining a coordinated To-Be state, while also showing that architecture value depends on how clearly governance, data, application, and technology decisions are translated into implementable controls.

The comparison indicates that prior studies contribute important but distinct elements. TOGAF-based higher-education studies primarily structure institutional domains, management services, and target-state blueprints [9], [10], [11], while REST-based studies demonstrate service interoperability between academic, learning, or library applications [12]–[13], [14]. Data-governance studies emphasize ownership, quality, and institutional responsibility [2], [5], but do not by themselves specify the service contract and operational workflow for repository reuse. The present study therefore does not claim novelty from TOGAF ADM or REST API in isolation. Its case-based contribution is the explicit linkage of (i) attribute-level ownership and mapping rules, (ii) an autonomy-preserving AIS–repository service boundary, (iii) implemented HTTPS/JWT access controls including 401 and 403 negative tests, and (iv) technical and paired workflow measurements. The remaining gap addressed is the limited empirical traceability between governance decisions, API implementation, synchronization outcomes, and end-to-end repository-entry work in one institutional case.

Accordingly, this study aims to (1) identify integration gaps between SIAKAD and the repository; (2) design a To-Be enterprise architecture using TOGAF ADM; (3) validate the architecture through stakeholder confirmation and external expert assessment; and (4) evaluate the implemented integration through functional, invalid-identifier, synchronization, consistency, API-response-time, and paired operational-time tests. The scientific contribution is a traceable case-based model in which data ownership is translated into an API contract and repository workflow, then assessed using both technical outcomes and operational observations. The practical contribution is a set of governance and implementation decisions for data authority, controlled access, mapping, synchronization, logging, and staged adoption. Because the study evaluates one institution and a limited set of controlled scenarios, it does not claim universal applicability or institution-wide performance.

2. RESEARCH METHOD

This research adopts a descriptive qualitative case-study approach strengthened by quantitative validation and operational measurement. Architectural requirements were derived from internal and technical documents, observation of the existing AIS–repository workflow, information from five key stakeholder roles, and analysis of application, data, and infrastructure conditions. These sources established the As-Is condition, identified overlapping academic attributes, defined data ownership, and guided the To-Be architecture. TOGAF ADM was applied from the Preliminary Phase through Phase E to structure architecture principles, vision, business architecture, information-systems architecture, technology architecture, and opportunities and solutions, consistent with recent phase-based EA applications [7], [9], [10], [11]. Evaluation was then conducted through four complementary evidence streams: formal stakeholder confirmation, independent external EA expert validation, controlled technical integration testing, and paired As-Is/To-Be operational-time measurement. This design separates architecture formulation from validation so that claims are supported by recorded stakeholder, expert, technical, and operational evidence rather than by conceptual projection alone.

Table 1. Key Informants

Code	Role	Main Information
I1	AIS Administrator	Academic master data and current system flow

p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

I2	Repository Administrator	Repository metadata and input process
I3	IT Unit Staff	Infrastructure and application integration
I4	Developer/Database Administrator	Database, API, and synchronization
I5	Academic Administrator	Operational process and user requirements

Table 2. TOGAF ADM Phases and Research Outputs

Phase	Main Activity	Research Output
Preliminary	Establish business, application, data, and technology principles	Architecture principles
Phase A	Define the integration vision, stakeholders, and value-chain scope	Integration vision and value-chain scope
Phase B	Analyze the As-Is workflow and design the To-Be business process	Target business process
Phase C	Analyze data ownership, data mapping, and application interaction	Data ownership, mapping, and integration architecture
Phase D	Define API, authentication, server, database, and communication requirements	Technology architecture
Phase E	Analyze architecture gaps, solution packages, and implementation priorities	Gap analysis, solution package, and implementation roadmap

Traceability from TOGAF ADM decisions to implementation and evaluation was maintained as follows. The Preliminary Phase established the principle that selected academic master data have one authoritative owner; Phase A bounded the scope to AIS–repository exchange; Phase B translated the repeated-entry problem into the To-Be workflow (Figures 3–4); Phase C assigned ownership and mapping rules to the eight candidate attributes (Table 8) and defined the application interaction; Phase D translated the service boundary into a REST endpoint, Bearer-token request configuration, HTTPS termination, proxy directives, and logging controls (Figures 5–7 and the technical configuration evidence); and Phase E prioritized integration, testing, and staged deployment (Tables 10, 12, 14, and 15). The evaluation evidence is linked to these decisions through the 20-record synchronization workbook, the 25-sample API performance artifacts, stakeholder/expert assessments, and paired operational-time observations. The public API path /v1 is supported by the cURL and environment evidence; however, the supplied Nginx screenshots show differing location/upstream configurations, so the internal proxy path is not treated as fully reconciled. This traceability demonstrates how the selected ADM phases informed the architecture and documented implementation, while unresolved deployment details and production-scale controls remain explicit limitations.

The architecture was evaluated in four complementary stages. First, formal stakeholder confirmation assessed whether the As-Is diagnosis and To-Be design reflected operational requirements. Second, two external Enterprise Architecture experts independently assessed architectural alignment, TOGAF application, domain architectures, data ownership, REST API design, security, roadmap feasibility, and overall consistency. Third, the implemented integration service was tested using predefined functional, negative-ID, synchronization, consistency, and response-time scenarios. Fourth, operational efficiency was measured using paired observations of the same academic-metadata retrieval and entry task under the As-Is and To-Be workflows. This layered design provides requirement-level, architecture-level, system-execution, and operational evidence.

2.1 Technical Evidence and Validation Criteria

Technical validity was assessed by linking each test objective to an acceptance criterion and recorded evidence. Functional testing covered 20 valid student identifiers and required HTTP 200 responses. Negative input testing covered five cases: four unregistered identifiers expected to return HTTP 404 and one malformed identifier expected to return HTTP 400. Security testing covered a valid token, a missing token, an invalid token, and a token with insufficient role privileges; the recorded negative outcomes were HTTP 401 for missing/invalid authentication and HTTP 403 for insufficient authorization. Synchronization testing covered 20 selected records, with mapping status and eight academic attributes audited per record. Performance was measured through 25 JMeter samples. These results describe the documented test runs and do not constitute a penetration test or a production-scale service-level assessment.

2.1.1 Technical Testing Environment

The technical evaluation used a local on-premises server with an Intel Core i7-11700 processor (8 cores), 16 GB DDR4 RAM, Ubuntu Server 22.04 LTS, and MySQL 8.0.32. The client used Windows 11 Pro, an Intel Core i5 processor, and 8 GB RAM, connected through a Gigabit LAN (1 Gbps) with a reported baseline latency below 1 ms. Postman was used for functional and security scenarios. The supplied Apache JMeter 5.5 Test Plan screenshot shows 25 users, a 5-second ramp-up, duration=60, and scheduler=false; the duration value therefore does not establish an active 60-second scheduler. The screenshot does not expose the loop-controller count and shows a fixed sampler path ending in student identifier 21010001. By contrast, the separate raw-log screenshot lists 25 requests for identifiers 21010001–21010025. Accordingly, the available screenshots do not establish that the displayed Test Plan generated the displayed raw log. The raw-log elapsed values transcribed from the screenshot range from 174 to 196 ms (sum 4,609 ms; arithmetic mean 184.36 ms). The separate HTML dashboard reports 25 samples, 0% errors, mean 183.12 ms, minimum 170 ms, maximum 194 ms, and throughput 5.46 requests/s. These values are retained as separate artifacts; the difference is unresolved. Machine-readable .jmx and .jtl files, complete assertions/timeouts, and a verifiable run-to-report link were not supplied, so exact independent reproduction remains incomplete. Both summaries are restricted to the documented 25-sample evidence and are not production-scale service-level guarantees.

Table 3. Technical Testing Environment

Component	Specification
Server type	Local on-premises server
Processor	Intel Core i7-11700, 8 cores
Memory	16 GB DDR4 RAM
Operating System	Ubuntu Server 22.04 LTS
DBMS	MySQL 8.0.32
Database	SIAKAD master database and Repository database
Network	Gigabit LAN
Network topology	Star
Bandwidth	1 Gbps
Baseline network latency	< 1 ms
Client OS	Windows 11 Pro
Client processor	Intel Core i5
Client memory	8 GB RAM
Testing tools	Postman and Apache JMeter
Number of requests	25 requests
Total response time	4,609 ms from the 25 visible raw-log elapsed values; HTML dashboard aggregate: 4,578 ms
Mean response time	184.36 ms (raw-log screenshot recalculation); 183.12 ms (HTML dashboard); discrepancy unresolved
Production API base URL	https://api.siakad.ac.id/v1
API resource	GET /students/{student_id}
Apache JMeter version and load	JMeter 5.5; screenshot shows 25 users, 5-second ramp-up, duration=60 with scheduler=false; fixed sampler ID 21010001; loop count not visible. Raw log lists 25 distinct IDs, so run-to-log correspondence is unverified.
JMeter dashboard result	Dashboard: 25 samples; 0% errors; mean 183.12 ms; min 170 ms; max 194 ms; 5.46 req/s. Separate raw-log screenshot: 25 HTTP 200 samples; mean 184.36 ms; min 174 ms; max 196 ms; 4,609 ms total. Same-run correspondence unresolved.
Execution date	14 August 2026

2.1.2 API Authentication, Transport Security, and Access Control

The confirmed public API endpoint is https://api.siakad.ac.id/v1, with the student resource GET /students/{student_id}. Requests use the Authorization: Bearer <token> header. The supplied authentication-policy artifact specifies JWT Bearer validation using RS256, issuer

https://siakad.ac.id/auth, audience https://api.siakad.ac.id/v1, a 3,600-second token lifetime, and required claims sub, iss, aud, exp, and roles. The integration service account is documented as sa_repository_sync with role ROLE_REPOSITORY_READER, permission READ_ACADEMIC_MASTER_DATA, and READ_ONLY access to approved academic attributes. Nginx screenshots show a TLS-enabled listener on port 443, TLS 1.2/1.3, certificate configuration, security headers, rate limiting, and reverse-proxy directives. However, the supplied screenshots show different location/upstream variants (/v1/ to an upstream /v1/ path in one artifact and /api/v1/ to a local upstream in another); the reconciliation spreadsheet proposes a /v1-to-/api/v1 mapping, but the screenshots do not consistently verify that exact deployed mapping. The public /v1 path is therefore used as the canonical client-facing endpoint, while the internal route remains a deployment detail requiring confirmation. A supplied cURL response confirms a successful HTTPS request with a valid certificate and HTTP 200. Separate response artifacts record HTTP 401 for a missing token and an invalid token, and an HTTP 403 response body identifies ROLE_STUDENT as insufficient for a route requiring ROLE_ADMIN. The 403 response body is available, but the corresponding original request-level export is not, so the authorization trace is less complete than the authentication evidence. The Python script supplied with the revision package is a prototype using an in-memory mock database and static placeholder-token comparison; it does not independently demonstrate RS256 signature verification or a production database connection. These artifacts support only the documented and tested cases; they do not constitute penetration testing or prove every permission combination.

2.1.3 Test Records, Attribute Mapping, and Synchronization Procedure

The technical workbooks contain 20 test records selected from the SIAKAD master database and related academic records. The documented inclusion criteria were: graduated academic status; coverage of five study programs in the Faculty of Science and Technology; variation in name length and characters; completeness of the eight mapped attributes without NULL values; and records associated with active final-project transactions. The records were selected as a controlled test set, not as a random or statistically representative sample. Student identifiers and names are omitted from the manuscript for privacy. The mapping comprises NIM (tb_mahasiswa.nim to eprint.creators_id), student name (tb_mahasiswa.nama_lengkap to eprint.creators_name), study program (tb_prodi.nama_prodi to eprint.department), faculty (tb_fakultas.nama_fakultas to eprint.division), final-project title (tb_ta.judul_ta to eprint.title), supervisor (tb_ta.dosen_pembimbing to eprint.advisors_name), graduation year (tb_yudisium.tahun_lulus to eprint.date_year), and academic status (tb_mahasiswa.status_akademik to eprint.is_validated). NIM is the lookup key; seven descriptive attributes are mapped to repository fields. Program and faculty values are standardized, supervisor strings are normalized, and academic status must be Lulus to activate the repository publication workflow. The title received from SIAKAD is treated as a draft; the repository retains control over the publication version. The audit workbook records source values and MATCH markers for all eight mapped attributes across 20 records (160 attribute-level checks), while the synchronization workbook records repository status as empty before synchronization and automatically populated afterward. These are workbook-based audit records, not a full database-dump comparison. The five negative-input scenarios comprise four unregistered identifiers (HTTP 404) and one malformed identifier (HTTP 400). Documented update rules include re-sync overwrite for SIAKAD-owned attributes, repository ownership of the final publication title, read-only repository fields for master attributes, and a versioned service contract. Downtime fallback, retry, and recovery policies are documented as architecture/operational rules but were not independently stress-tested in the supplied run.

2.1.4 Postman and Apache JMeter Test Configuration

The supplied Postman screenshots document a valid student request, an unregistered identifier, and Bearer-token header configuration. Separate response artifacts record HTTP 401 for a missing token, HTTP 401 for an invalid token, and an HTTP 403 response body for insufficient role privileges. The corrected JSON collection included in the revision package is a normalized test template assembled from the supplied endpoint and scenario documentation; it is not represented as the original machine-readable Postman execution export. The public endpoint confirmed by the cURL and environment evidence is https://api.siakad.ac.id/v1/students/{student_id}. Earlier supporting screenshots and the service-account permission record contain /api/v1, while other Nginx and client-facing artifacts show /v1. The reconciliation spreadsheet proposes that Nginx maps public /v1/ requests to an internal

/api/v1/ route, but the supplied Nginx screenshots show differing location/upstream configurations and do not consistently establish that mapping. The public /v1 path is therefore standardized in the manuscript, while internal route equivalence remains unverified. The JMeter 5.5 Test Plan screenshot shows 25 users, a 5-second ramp-up, duration=60 with scheduler=false, and a fixed sampler identifier; the raw-log screenshot instead lists 25 distinct identifiers. The screenshot does not show the loop-controller count, and the available artifacts do not establish that this Test Plan generated that raw log. The HTML dashboard reports 25 samples, zero errors, mean 183.12 ms, a 170–194 ms range, and 5.46 requests/s; recalculation from the separate raw-log screenshot gives 184.36 ms (174–196 ms). Native machine-readable .jmx/.jtl files and a complete execution-to-report chain were not supplied, so these artifacts support a limited documented test result rather than full reproducibility.

2.1.5 Postman and Apache JMeter Test Configuration

The operational comparison pairs the same 25 student identifiers and the same terminal output: a repository record saved with the required academic metadata. The operational protocol defines the start event as the beginning of record-specific processing and the end event as successful saving of that repository record. In As-Is, the operator retrieves the record in SIAKAD, copies eight attributes, transfers them to the repository form, checks the fields, and saves. In To-Be, the operator enters the same NIM in the repository form, retrieves and auto-populates the approved attributes through the API, reviews the result, and saves. Thus, the workflow actions differ by design, but the identifier and saved repository-record outcome are held constant. Five operators (O1–O5), corresponding to five operational roles, each completed five paired observations; the same operator performed both conditions for each assigned record. Timestamped observations were recorded on 14 August 2026. All As-Is observations were conducted first (08:30–09:19 WIB), followed by all To-Be observations (10:30–10:54 WIB). The order was not randomized or counterbalanced, and no formal familiarization or washout record was supplied. The workbook records start/end timestamps, but no screen-level event trace independently verifies the exact click or event used to start and stop each timer. Pairing controls stable operator differences within each record pair, but does not eliminate order or learning effects. The Shapiro–Wilk test on paired differences gave W=0.985 and p=0.962; independence of differences, particularly because each operator contributed five pairs, was not separately tested with a clustered model. The paired t-test is therefore interpreted as a case-specific within-record comparison, not a fully controlled causal estimate.

2.2 Formal Stakeholder Confirmation

Formal stakeholder confirmation involved five operational representatives: AIS Administrator (I1), Repository Administrator (I2), IT Unit Staff (I3), Developer/Database Administrator (I4), and Academic Administrator (I5). Ten instrument items (SV1–SV10) were rated on a five-point Likert scale. The supplied workbook contains the respondent-by-item score matrix and concise item labels. Item means were SV1=4.60, SV2=4.60, SV3=5.00, SV4=4.40, SV5=5.00, SV6=4.80, SV7=5.00, SV8=4.40, SV9=4.60, and SV10=5.00. Respondent means were I1=4.80, I2=4.80, I3=4.70, I4=4.70, and I5=4.70; the overall mean was 4.74/5.00. The scale defines 4.21–5.00 as Very Feasible/Accept, 3.41–4.20 as Feasible, 2.61–3.40 as Moderately Feasible, 1.81–2.60 as Less Feasible, and 1.00–1.80 as Not Feasible. The reported mean is descriptive stakeholder confirmation and is not presented as psychometric scale validation. The supplied workbook does not contain expanded full-sentence questionnaire prompts beyond the concise item labels, so the exact questionnaire form should be attached separately if available.

Table 4. Formal Stakeholder Confirmation Results

Item	Validation Statement	Scores (I1-I5)	Mean
SV1	Accuracy of the As-Is diagnosis	5, 5, 4, 4, 5	4.60
SV2	Relevance of the operational problems	5, 5, 5, 4, 4	4.60
SV3	Suitability of the To-Be workflow	5, 5, 5, 5, 5	5.00
SV4	Clarity of data ownership	4, 4, 4, 5, 5	4.40
SV5	SIAKAD as the authoritative data source	5, 5, 5, 5, 5	5.00
SV6	REST API interoperability	5, 5, 5, 5, 4	4.80
SV7	Reduction of redundant data entry	5, 5, 5, 5, 5	5.00

p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

SV8	Implementation feasibility	4, 4, 4, 5, 5	4.40
SV9	Data consistency and traceability	5, 5, 5, 4, 4	4.60
SV10	Fulfillment of integration requirements	5, 5, 5, 5, 5	5.00
Overall	**Mean of all ten items**	—	**4.74/5.00**

2.3 External Enterprise Architecture Expert Validation

External Enterprise Architecture validation involved two anonymized experts. Expert 1 is documented as holding a doctoral qualification in Information Systems/Computer Science, TOGAF 9 certification, and more than 12 years of EA practice and academic experience. Expert 2 is documented as holding a master's qualification in Information Systems/Information Technology, TOGAF Enterprise Architect Practice and CISA credentials, and more than 10 years of IT governance, audit, and infrastructure experience. The experts independently rated twelve criteria (EV1–EV12) on a five-point scale. The workbook supplies criterion-level scores, individual means of 4.75/5.00 and 4.58/5.00, and a combined mean of 4.67/5.00. The category thresholds define 4.21–5.00 as Very Good/Very Feasible, 3.41–4.20 as Good/Feasible, 2.61–3.40 as Fair, 1.81–2.60 as Poor, and 1.00–1.80 as Very Poor. The lower mean criteria were security/authentication adequacy and implementation-roadmap feasibility (4.00 each), indicating areas for refinement. The assessment represents expert judgment, not independent penetration testing or production-readiness certification. The exact full-text instrument wording is not included in the supplied workbook, although the criterion labels and individual scores are available.

Table 5. External Enterprise Architecture Expert Validation Results

Criterion	Expert 1	Expert 2	Mean
EV1-Problem–architecture vision alignment	5	5	5.00
EV2-Appropriateness of TOGAF ADM application	5	4	4.50
EV3-As-Is/To-Be architecture consistency	5	5	5.00
EV4-Business architecture	5	4	4.50
EV5-Data architecture	5	5	5.00
EV6-Application architecture	5	5	5.00
EV7-Technology architecture	4	5	4.50
EV8-Master-data ownership	5	4	4.50
EV9-REST API integration design	5	5	5.00
EV10-Security and authentication adequacy	4	4	4.00
EV11-Implementation roadmap feasibility	4	4	4.00
EV12-Overall architectural consistency	5	5	5.00
Overall mean	4.75	4.58	4.67

2.4 Operational-Time Efficiency Measurement

Operational-time efficiency was calculated from 25 paired observations (Obs-01–Obs-25) recorded on 14 August 2026. Five operators (O1–O5) each completed five pairs, with the same operator carrying out the As-Is and To-Be task for each record. The workbook includes operator role codes and start/end timestamps for both conditions. The As-Is sequence preceded the To-Be sequence for all operators; no randomization or counterbalancing was documented. The equivalent outcome was a saved repository record containing the same academic metadata for the same student identifier. As-Is required opening SIAKAD, locating the record, copying the eight attributes, opening the repository form, entering and checking the fields, and saving. To-Be required opening the repository form, entering the NIM, retrieving and auto-populating the attributes through the authenticated API, reviewing, and saving. The percentage reduction is calculated as [(mean As-Is − mean To-Be)/mean As-Is] × 100, and the paired-samples t-test is applied to within-record differences. Shapiro–Wilk was used to assess normality of the differences. Since the fixed As-Is-first order may create familiarity effects and each of the five operators contributed multiple pairs, possible order effects and within-operator dependence cannot be ruled out; no cluster-adjusted model was fitted. Operator-level mean paired differences ranged from 79.36 to 85.34 seconds; this descriptive range shows the direction of change across the

five operators but does not replace a cluster-adjusted analysis. The findings are therefore reported as a case-specific operational comparison.

3. RESULTS AND DISCUSSION

The architecture analysis is presented sequentially according to the selected TOGAF ADM phases, beginning with architecture principles and vision, followed by business, information-systems, and technology architectures, and concluding with opportunities, solutions, technical evaluation, and implementation considerations.

3.1 Preliminary Phase

The Preliminary Phase establishes the architecture principles that guide all subsequent decisions. Four principles are defined to prevent the future integration solution from creating another isolated application or unmanaged duplicate database.

Table 6. Architecture Principles

Principle	Statement	Expected Result
Business	Integration must support academic and repository processes without unnecessary duplication.	A streamlined cross-system workflow aligned with academic-service needs.
Application	Applications remain functionally autonomous but expose controlled services for interoperability.	AIS and repository communicate through defined API services rather than manual re-entry.
Data	Academic master data must have a clear authoritative source, standardized format, and controlled synchronization.	Consistent student and graduation information across systems.
Technology	Integration must use secure, maintainable, standards-based infrastructure.	Authenticated HTTPS REST API, auditable service access, and maintainable deployment.

3.2 Architecture Vision

The architecture vision is to establish an integrated academic-information environment in which validated academic master data can be reused by the repository without requiring users to re-enter the same information. The target state emphasizes interoperability, consistency, traceability, and controlled access rather than the physical merger of the two applications. The main value chain covers academic-data management in the AIS, final-project metadata submission, identity validation, repository ingestion, and document publication. The stakeholders include academic administrators, AIS administrators, repository administrators, the institutional IT unit, developers/database administrators, lecturers or supervisors, and students as service users. Their shared objective is to reduce repeated data handling while maintaining clear ownership of academic and repository-specific information.

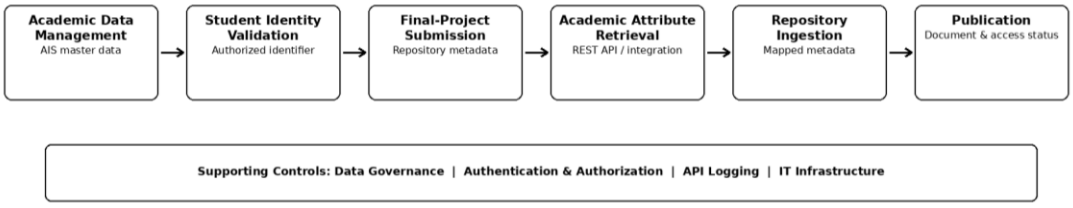

Figure 1. Architecture Vision value chain for AIS–repository integration

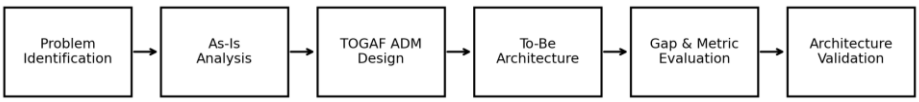

Figure 2. Research workflow from problem identification to technical validation

3.3 Business Architecture

The As-Is workflow is characterized by separate application processes. Academic data are maintained in the AIS, while repository metadata are entered through the repository interface. When overlapping attributes are required, users or operators may copy or retype information from one context to another. This creates manual hand-offs between applications and introduces opportunities for spelling differences, outdated values, and repeated verification.

Vol. 11, No. 2, Oktober 2026, Page. 621-643
https://journal.uin-alauddin.ac.id/index.php/instek
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

Table 7. As-Is Integration Problems

Problem	As-Is Condition	Operational Impact
Repeated entry	Academic attributes are entered again in the repository.	Additional processing time and human error risk.
Fragmented applications	AIS and repository do not exchange data directly in the controlled scenario.	Users become manual intermediaries.
Inconsistent metadata	Names, study programs, supervisors, or graduation data may use different formats.	Lower data quality and repeated verification.
Independent updates	Changes in AIS are not automatically reflected in repository metadata.	Potentially outdated repository information.

The To-Be business process removes unnecessary re-entry. A repository user provides or selects a student identifier, after which the repository requests authorized academic attributes from the AIS through the integration service. The returned data are validated and mapped automatically, while repository-specific fields such as abstract, keywords, document file, access status, and publication metadata remain managed by the repository. The proposed process therefore reduces manual hand-offs while preserving functional separation between systems.

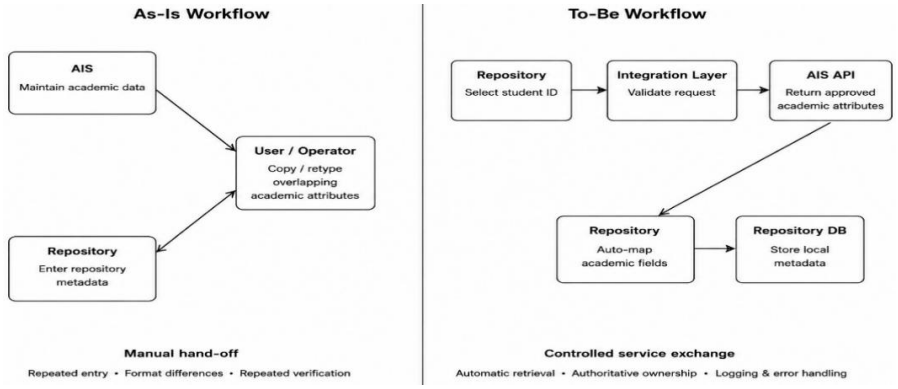

Figure 3. Detailed As-Is and To-Be AIS-repository business workflow

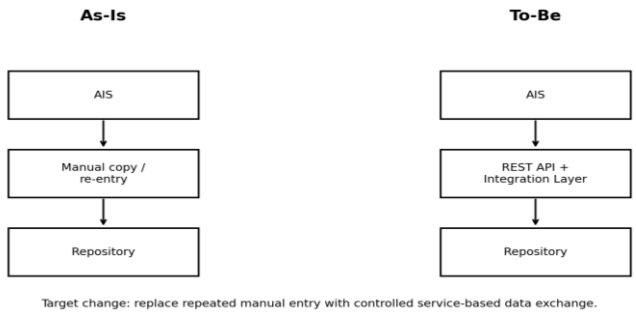

Figure 4. Comparison of the As-Is and To-Be workflows

3.4 Information Systems Architecture

The same real-world entity, such as a student or supervisor, can therefore be represented independently in both databases. Eight attributes are used in the initial mapping to identify the main redundancy risk.

Table 8. Data Mapping and Ownership

Attribute	AIS	Repository	Target Ownership / Rule
Student ID	Available	Used	AIS master; repository references value
Student name	Available	Used	AIS master; retrieved through API
Study program	Available	Used	AIS master; standardized code/name
Faculty	Available	Used	AIS master; retrieved through API
Final-project title	Potentially available	Primary metadata	Repository manages publication version; validation rule required

Supervisor	Potentially available	Used	AIS/authorized academic source; normalized format
Graduation year	Available	Used	AIS master
Graduation status	Available	Required for validation	AIS master

The To-Be data architecture establishes the AIS as the source of truth for selected academic master data. The repository does not directly access the AIS database. Instead, an API service exposes only approved attributes and applies validation, authentication, and mapping rules. This service boundary reduces tight coupling and enables both systems to evolve independently. The proposed application flow is: AIS Database → REST API Service → Authentication/Integration Layer → Repository Application → Repository Database. A representative service is GET /v1/students/{student_id}. The service returns only the attributes required for repository processing, while repository-specific metadata remain local to the repository.

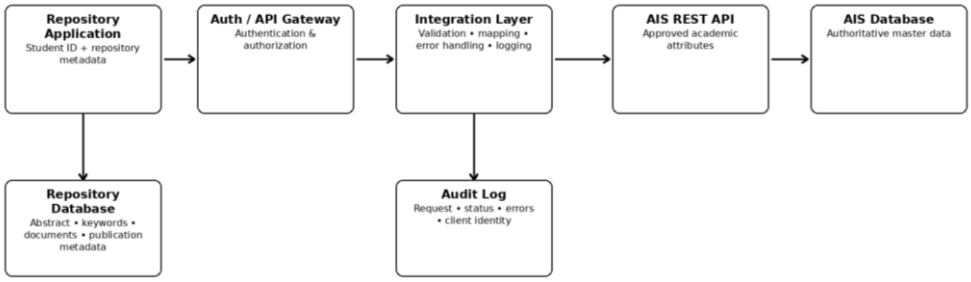

Figure 5. Target application and data integration architecture

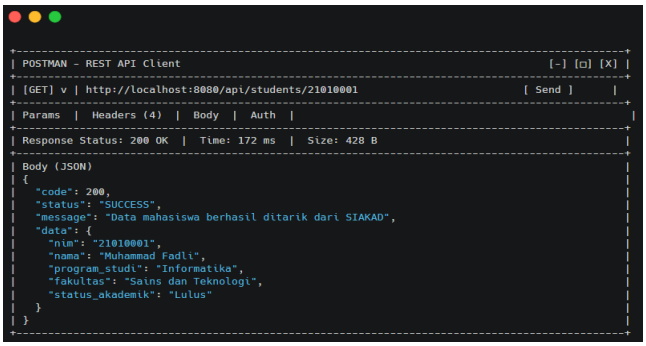

Figure 6. REST API test with a valid student-data response

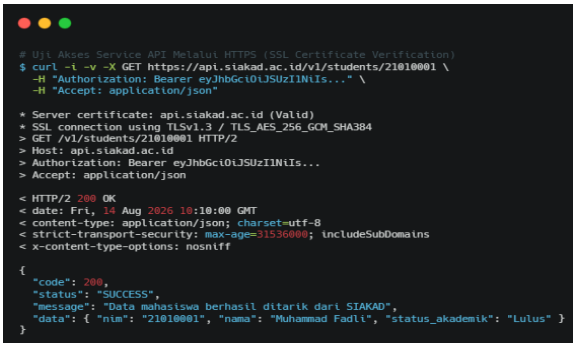

Figure 7. Successful production HTTPS request to the confirmed REST API endpoint (HTTP 200)

3.5 Technology Architecture

The target technology architecture uses a service-based integration pattern with the confirmed public base URL https://api.siakad.ac.id/v1 and the student resource GET /students/{student_id}. The available Nginx configuration excerpts show HTTP-to-HTTPS redirection, TLS termination on port 443, TLS 1.2/1.3, certificate configuration, security headers, rate limiting, and reverse-proxy directives. However, the supplied excerpts show different location/upstream path variants; the reconciliation spreadsheet proposes a public /v1/ to internal /api/v1/ mapping, but this exact mapping is not consistently demonstrated by the screenshots. Accordingly, the manuscript treats /v1 as the

confirmed public client-facing path and does not claim that the internal upstream route has been conclusively reconciled. The Postman environment parameterizes Bearer JWT credentials; the documented validation policy specifies RS256, issuer/audience checks, required claims, and a 3,600-second token lifetime. The service account sa_repository_sync is assigned ROLE_REPOSITORY_READER with READ_ACADEMIC_MASTER_DATA permission and READ_ONLY access. Executed negative security scenarios recorded HTTP 401 for missing and invalid tokens and HTTP 403 when a ROLE_STUDENT token accessed an endpoint requiring ROLE_ADMIN. A valid request returned HTTP 200. The supplied Python file is a mock-data prototype and is not proof of production JWT cryptographic validation. These artifacts support the documented security cases but do not constitute a penetration test or prove every possible access-control path. The integration layer is designed to validate identifiers, map approved attributes, and record execution outcomes; SIAKAD and repository databases remain separate.

Table 9. Target Technology Components

Component	Function
REST API Service	Provides controlled access to approved AIS academic data.
Authentication/Authorization	Restricts service access to trusted repository clients.
Integration Layer	Validates requests, maps attributes, handles errors, and records logs.
HTTPS	Protects data in transit.
AIS Database	Authoritative source for selected academic master data.
Repository Database	Stores repository-specific metadata and document references.

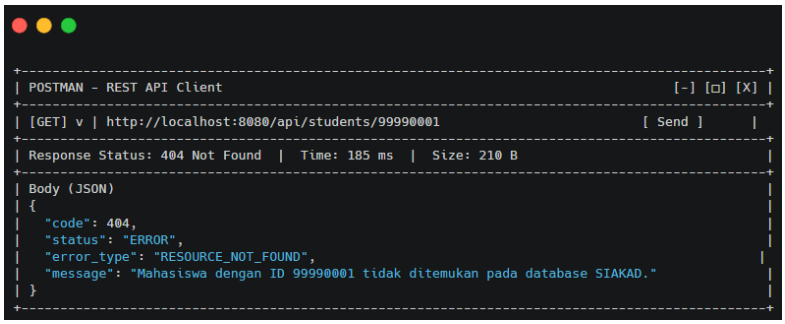

Figure 8. Invalid student-identifier handling with an HTTP 404 response

3.6 Opportunities and Solutions

Phase E compares the As-Is and To-Be architectures and translates the proposed changes into measurable benefits. The solution package consists of academic-data ownership rules, a standardized data mapping, an authenticated REST API, an integration layer, repository-side automatic field population, and logging for synchronization and error tracing.

Table 10. Gap Analysis and Solutions

As-Is	Gap	To-Be Solution
Separate applications	No controlled data exchange	REST API-based integration
Repeated academic input	Redundancy and human error	Automatic retrieval from AIS
Independent data formats	Potential inconsistency	Standard mapping and master-data rules
Manual verification	Additional operational steps	Automated validation and service response
Limited traceability	Difficult to identify exchange failures	API logging and error handling

3.7 Stakeholder Confirmation, External Expert Validation, and Operational Efficiency

Stakeholder confirmation yielded an overall mean of 4.74/5.00 across five operational representatives. The newly supplied SV1–SV10 response matrix permits item-level verification: means ranged from 4.40 to 5.00, with SV3, SV5, SV7, and SV10 each scoring 5.00 and SV4 and SV8 scoring 4.40. Respondent-level means were 4.80, 4.80, 4.70, 4.70, and 4.70. The results indicate positive acceptance among the five participating roles, but are not interpreted as psychometric validation or institution-wide acceptance.

Table 11. Operational-Time Efficiency Measurement Results

Metric	As-Is	To-Be / Result	Improvement / Statistical Evidence
Paired observations (n)	25	25	Same/comparable task pairs
Mean processing time	94.60 s	11.41 s	87.94% reduction

Median processing time	94.00 s	11.30 s	Descriptive statistic
Standard deviation	6.57 s	0.96 s	Sample standard deviation
Observed range	83–110 s	9.8–14.0 s	Observed minimum–maximum
Mean paired difference	-	83.19 s	95% CI [80.87, 85.51] s
Paired-samples t-test	-	-	t(24)=74.05; p<0.001
Normality of paired differences	-	-	Shapiro–Wilk W=0.985; p=0.962
Manual cross-system hand-off	1 required	0 manual	Eliminated in the evaluated workflow
Duplicated-data-entry fields	8 fields	0 fields	Eliminated in the evaluated workflow

External expert validation produced mean scores of 4.75/5.00 for Expert 1 and 4.58/5.00 for Expert 2, with a combined mean of 4.67/5.00 across twelve criteria. The criterion-level ratings in Table 5 show the lowest means for security/authentication and implementation-roadmap feasibility (4.00 each). Expert recommendations included stronger authentication design, transition planning, and a fallback/read-only mechanism when the primary AIS service is unavailable. These recommendations are treated as refinement actions; the expert score does not substitute for a security test or production-readiness assessment.

3.8 Technical Evaluation of the Integration Service

Technical evaluation covered functional, negative-input, authentication, authorization, synchronization, consistency, and performance scenarios documented in the Postman materials, OpenAPI contract, Excel workbooks, SQL snapshot, and JMeter evidence. The supplied records show 20/20 valid student requests with HTTP 200; five negative inputs handled as expected (four unregistered identifiers with HTTP 404 and one malformed identifier with HTTP 400); two authentication-denial scenarios with HTTP 401 (missing and invalid token); and one insufficient-role response artifact with HTTP 403. The synchronization workbook records 20/20 records as mapped and automatically populated. Its eight-attribute audit records 160 MATCH markers across NIM, name, study program, faculty, final-project title, supervisor, graduation year, and academic status. The SQL snapshot lists 20 source records and 20 after-sync repository records, but its displayed after-sync query omits supervisor and graduation-year fields; it therefore corroborates record-level population but does not independently show all eight fields in the SQL output. The workflow sheet reports repository status as empty before synchronization and automatically populated afterward. The public endpoint used in the manuscript is https://api.siakad.ac.id/v1; internal Nginx route equivalence remains unresolved because configuration excerpts differ. These results describe the supplied controlled test evidence and are not a full production database audit.

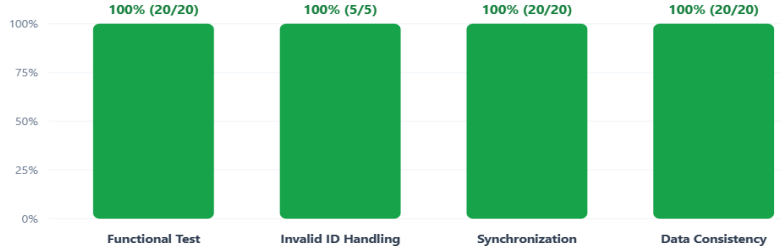

Figure 9. Technical integration test success rates

Table 12. Technical Integration Test Results

Test metric	Test scope	Result	Status
Valid API retrieval	20 valid student identifiers	20/20 HTTP 200 (100%)	PASS
Unregistered identifier	4 unregistered identifiers	4/4 HTTP 404	PASS
Malformed identifier	1 malformed NIM format	1/1 HTTP 400	PASS
Missing Bearer token	1 security scenario	HTTP 401 (MISSING_BEARER_TOKEN)	PASS
Invalid Bearer token	1 security scenario	HTTP 401 (INVALID_TOKEN)	PASS
Insufficient role privilege	ROLE_STUDENT token to ROLE_ADMIN endpoint	HTTP 403 (INSUFFICIENT_PRIVILEGES)	PASS

Synchronization	20 student records	20/20 mapped and auto-populated	PASS
Eight-attribute audit	20 records × 8 attributes (160 entries)	160 entries marked MATCH; repository status empty before and auto-filled after	RECORDED PASS
API response time	25 JMeter samples	Dashboard: mean 183.12 ms (170–194 ms); separate raw-log screenshot: mean 184.36 ms (174–196 ms); both display 25 successful samples, but same-run correspondence is unverified.	MEASURED ARTIFACTS; discrepancy and run linkage unresolved

For the functional scenario, 20 valid student identifiers returned HTTP 200. Five negative input tests were handled deterministically: four unregistered identifiers returned HTTP 404 with RESOURCE_NOT_FOUND, while one malformed identifier returned HTTP 400 with INVALID_INPUT_FORMAT. Authentication tests recorded HTTP 401 for both a missing Bearer token and an invalid token. The authorization response artifact records HTTP 403 when a valid ROLE_STUDENT token attempted to access an endpoint requiring ROLE_ADMIN; the original request-level export is unavailable. For synchronization, the workbook marks all 20 records as MATCH/Mapped and PASS, and the eight-attribute audit records 160 MATCH markers across 20 records. The SQL snapshot lists 20 after-sync records but its displayed query omits advisors_name and date_year, so it does not independently verify all eight fields at the database-output level. The before-sync SQL section uses a NULL placeholder row while also stating that zero rows were returned; this should be interpreted as a schematic empty-state illustration, not a literal result set. These records document the supplied run and status labels; they do not constitute a full database dump or byte-for-byte comparison of complete database states.



Figure 10. REST API response-time measurements for 25 requests

Target: https://api.siakad.ac.id/v1/students/{student_id}
Execution date: 14 August 2026 | JMeter 5.5 | 25 samples

Metric	Result	Metric	Result
Samples	25	Successful	25 / 25 (100%)
Errors	0 (0.00%)	Mean response	183.12 ms
Minimum	170 ms	Maximum	194 ms
Throughput	5.46 req/s	Audit status	PASS

Values transcribed from the supplied jmeter_report_index.html.

Figure 11. Apache JMeter HTML dashboard summary for 25 REST API requests

API response-time performance was assessed using the supplied JMeter evidence dated 14 August 2026. The raw-log screenshot contains 25 entries, each with response code 200 and success=true. Recalculation from the 25 visible elapsed values gives a total of 4,609 ms, a minimum of 174 ms, a maximum of 196 ms, and an arithmetic mean of 184.36 ms. The separate HTML dashboard reports 25 samples, zero errors (0.00%), mean response time 183.12 ms, minimum 170 ms, maximum 194 ms, and throughput 5.46 requests/s (a displayed aggregate total of 4,578 ms). Because the two summaries differ, and because the Test Plan screenshot shows a fixed student identifier whereas the raw log lists 25 distinct identifiers, the available materials do not establish that the screenshots represent the same execution. Both summaries are therefore reported separately and remain unresolved pending the native machine-readable .jmx/.jtl files and their execution metadata. The Test Plan screenshot specifies

p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

JMeter 5.5, 25 virtual users, a 5-second ramp-up, duration=60, and scheduler=false; the loop-controller count is not visible. The evidence supports two separately documented 25-sample success summaries, not a verified 60-second scheduled load test or a reproducible load profile. The results are limited to the controlled samples supplied.

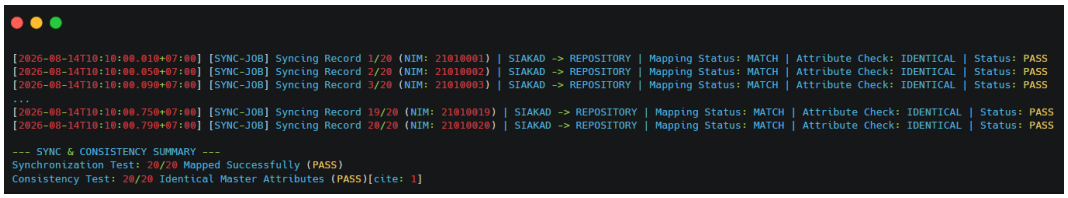

Figure 12. Data synchronization and consistency execution log for 20 records

The synchronization documentation records 20 processed records and labels the outcome as 20/20 matched and automatically populated. The newly supplied SQL snapshot lists 20 source records and 20 repository records after synchronization. Its before-sync section shows a schematic NULL placeholder row while stating that zero rows were returned; it is therefore not a literal query result and should be read only as an illustration of the empty repository state. The after-sync output displays 20 records but omits advisors_name and date_year from the selected columns, although both fields are included in the eight-attribute mapping. Consequently, the SQL snapshot does not demonstrate all eight attributes for all 20 records. The separate workbook audit records MATCH markers for NIM, student name, study program, faculty, final-project title, supervisor, graduation year, and academic status across 20 records (160 attribute-level entries); this is a workbook-reported consistency audit rather than a full database-level before/after comparison. The conflict-policy sheet documents SIAKAD-authoritative updates, repository control of final publication titles, read-only master fields, and versioned service-contract principles. Retry, rollback, and recovery after a simulated outage were not independently executed in the supplied evidence.

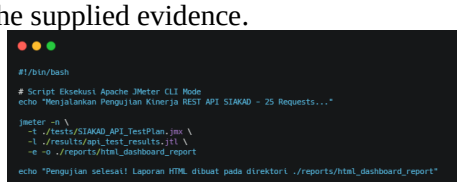

Figure 13. Apache JMeter non-GUI execution and HTML report-generation command

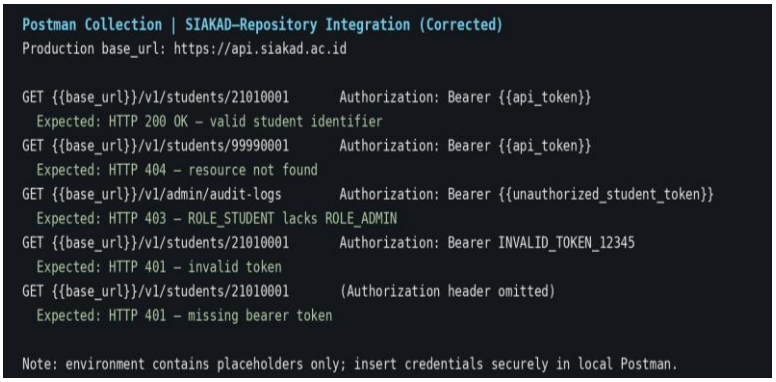

Figure 14. Corrected Postman collection excerpt using the confirmed production endpoint



Vol. 11, No. 2, Oktober 2026, Page. 621-643
https://journal.uin-alauddin.ac.id/index.php/instek
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

Figure 15. OpenAPI 3.0.3 production server definition using https://api.siakad.ac.id/v1



Figure 16. Nginx configuration excerpts showing HTTPS, reverse-proxy directives, and differing public/internal path variants



Figure 17. Example JSON payload for a successful HTTP 200 response

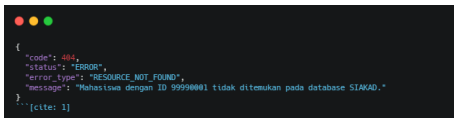

Figure 18. Example JSON payload for an invalid identifier and HTTP 404 response

Additional security evidence is provided to distinguish token authentication from role-based authorization and to document the service account used by the repository integration client.
The role-based negative test returned HTTP 403 when a valid token associated with ROLE_STUDENT attempted to access an administrative endpoint requiring ROLE_ADMIN. This complements the HTTP 401 tests for missing and invalid tokens and distinguishes authentication failure from authorization denial.



Figure 19. Role-based authorization denial for a token with insufficient privileges (HTTP 403)

The service-account record identifies the integration client as sa_repository_sync, assigns ROLE_REPOSITORY_READER, and limits access to READ_ACADEMIC_MASTER_DATA with READ_ONLY permission. This documents the intended service identity and least-privilege policy; the authorization response artifact separately demonstrates rejection of a token whose role does not satisfy the endpoint requirement. The service-account screenshot labels the permission endpoint as /api/v1. A reconciliation spreadsheet proposes that the public /v1/ path be mapped by Nginx to an internal /api/v1/ route, but the supplied Nginx screenshots show differing location/upstream configurations and do not consistently verify that exact mapping. The public endpoint is therefore standardized as /v1 in the manuscript, while internal route equivalence remains an item requiring confirmation against the active deployment configuration.

Vol. 11, No. 2, Oktober 2026, Page. 621-643
https://journal.uin-alauddin.ac.id/index.php/instek
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI



Figure 20. Repository integration service-account role and read-only permission

3.9 Application and Data Interaction Architecture

Table 13. Application and Data Interaction Matrix

Source	Target	Data / Service	Mechanism	Control
AIS	Integration Layer	Student identity, program, faculty, graduation status	REST API	Authentication + authorization
Integration Layer	Repository	Validated academic attributes	JSON/HTTPS	Schema validation + mapping
Repository	Repository DB	Abstract, keywords, document metadata	Internal service	Repository business rules
Integration Layer	Audit Log	Request, status, error information	Logging	Timestamp + client identity

The interaction matrix shows that the integration layer acts as the service boundary between the AIS and the repository. This separation is important because the repository does not depend on direct access to the AIS database. Instead, data exchange is performed through controlled services with authentication, validation, mapping, and logging.

3.10 Implementation Roadmap and Risks

Table 14. Implementation Priority Roadmap

Stage	Main Activity	Output	Priority
1	Confirm data ownership and attribute mapping	Approved data dictionary	High
2	Develop authenticated AIS API	Academic-data service	High
3	Implement repository integration layer	Automatic field population	High
4	Conduct functional, synchronization, consistency, error-handling, and response-time testing	Verified controlled integration-test results	High
5	Conduct architecture review and implementation-readiness assessment	Reviewed architecture blueprint and readiness actions	Medium
6	Operational deployment and monitoring	Production integration and logs	Medium

Table 15. Risk and Mitigation Matrix

Risk	Potential Impact	Mitigation
Unauthorized API access	Exposure of academic data	Authentication, authorization, HTTPS, audit logging
Schema mismatch	Failed synchronization	Versioned API contract and schema validation
Incorrect master-data mapping	Inconsistent repository metadata	Approved data dictionary and mapping tests
AIS service downtime	Repository cannot retrieve master data	Retry, timeout, caching policy, fallback procedure
Uncontrolled changes	Integration failure after application updates	Change management and regression testing

3.11 Discussion

3.11.1 Architecture Findings

The central architectural finding is that AIS–repository integration can be achieved without database consolidation or removal of specialized functions. The proposed design assigns selected academic master data to the AIS as the authoritative source, while repository-specific publication metadata remains under repository control. This governance rule is implemented through a controlled service boundary rather than direct database access. The result addresses the actual As-Is problem, namely the

p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

absence of a governed mechanism for reusing authoritative academic attributes, rather than treating the existence of two applications as the problem itself. This interpretation is consistent with higher-education data-governance studies emphasizing data quality, synchronization, and clearly assigned institutional responsibilities [2], [5], while repository research shows that repository platforms retain distinct software, metadata, and maintenance requirements [6].

From the data-architecture perspective, the authoritative-source rule identifies which system controls each shared attribute. From the application-architecture perspective, the REST API and integration layer decouple repository functions from the internal AIS database while providing validation and mapping. From the technology perspective, HTTPS, authentication, explicit error handling, and logging provide controls for secure and traceable exchange. These decisions translate the TOGAF blueprint into implementation-oriented controls rather than leaving the target architecture at the conceptual level [4], [11], [15].

The implementation trace can be followed from requirements to evidence and is summarized in Table 17. The need to eliminate repeated entry is addressed by the Phase B To-Be workflow and Figures 3–4; attribute ownership and eight field mappings are specified in Phase C and Table 8; the service contract, HTTPS transport, Bearer-token authentication, and service-account permission are assigned to Phase D and the implementation configuration; synchronization and consistency are represented by the 20-record workbook and 160 attribute-level MATCH markers; input and security handling are represented by the HTTP 400/401/403/404/200 response artifacts; and operational benefit is assessed using 25 paired observations in Table 11 and Appendix A. The chain links architecture requirements and decisions to implementation components and evaluation evidence. The route-version discrepancy in one permission artifact, the absence of native JMeter .jmx/.jtl files, and the absence of independent database snapshots remain evidence limitations.

3.11.2 Validation and Evaluation

The validation results provide complementary but distinct evidence. Stakeholder confirmation produced an overall mean of 4.74/5.00 across five operational roles, with all ten SV1–SV10 item means available (4.40–5.00). Two external EA experts produced a mean of 4.67/5.00 across twelve criteria; the lowest criteria were security/authentication adequacy and roadmap feasibility (4.00/5.00). The expert profiles and criterion-level scores are documented in the supplied workbook. These scores represent stakeholder acceptance and expert architectural judgment, not independent proof of security, usability, or production readiness.

The technical tests provide evidence for the documented scenarios: 20/20 valid requests returned HTTP 200; four unregistered identifiers returned HTTP 404; one malformed identifier returned HTTP 400; missing and invalid tokens returned HTTP 401; and a separate response artifact records HTTP 403 when a token with ROLE_STUDENT lacks the required ROLE_ADMIN privilege. Twenty records were marked synchronized, and the workbook audit records 160 attribute entries across eight fields as MATCH. The HTML dashboard reports a mean API response time of 183.12 ms across 25 samples, with a 0% error rate and a 170–194 ms range. Recalculation from the separate raw-log screenshot gives 184.36 ms (174–196 ms); the relationship between these two artifacts is unresolved. API response time is a technical service-latency measure, whereas the 94.60 s to 11.408 s result measures the complete operational task; the two measures have different boundaries and should not be conflated. The missing-token and invalid-token response artifacts support authentication rejection; the 403 body supports the authorization outcome, although its original request-level trace is not included. The SQL after-sync output omits two mapped fields, and the Python code is a mock-data prototype rather than production integration source. These tests do not establish performance under sustained production traffic.

Across the 25 paired observations, mean processing time decreased from 94.60 s (SD=6.57) in the As-Is workflow to 11.408 s (SD=0.965) in the To-Be workflow, a reduction of 87.94%. Median times were 94.00 s and 11.30 s, respectively; observed ranges were 83–110 s for As-Is and 9.8–14.0 s for To-Be. The mean paired difference was 83.192 s (SD=5.618; 95% CI [80.87, 85.51]), and the paired-samples t-test was statistically significant, $t(24)=74.05$, $p<0.001$. The Shapiro–Wilk test on paired differences gave $W=0.985$ and $p=0.962$, providing no evidence against normality of the difference distribution. Descriptive mean paired differences across operators ranged from 79.36 to 85.34 s (O1=84.12; O2=82.68; O3=84.46; O4=79.36; O5=85.34). Five operators each contributed five paired observations, and the same operator performed both conditions within each pair. However, all As-Is

observations preceded all To-Be observations, so familiarity and order effects may have influenced the second condition. The repeated contribution of five operators also means within-operator dependence cannot be ruled out; no cluster-adjusted model was fitted. The 87.94% reduction is therefore an observed case-specific association under the recorded sequence, not a causal estimate isolated from learning or operator effects.

3.11.3 Practical Implications

The practical implication is that technical interoperability must be accompanied by organizational interoperability. An institution adopting this architecture needs formal rules for who may modify authoritative attributes, who approves repository access to AIS services, how service accounts and credentials are governed, which unit owns incident response and service-level monitoring, and how schema changes are coordinated. The roadmap and risk matrix therefore remain essential implementation components. Authentication and HTTPS, API logging, schema validation, versioned service contracts, retry and fallback policies, change management, and regression testing should be treated as governance controls rather than optional technical details.

The architecture is most likely to be reproducible in institutions that have a stable academic master-data source, identifiable ownership of shared attributes, a repository that can consume REST/HTTPS services, an integration-capable IT team, and infrastructure that can support authenticated service communication and logging. Replication also requires comparable operating procedures, agreement on which attributes remain local to the repository, controlled API credentials, change-management arrangements, and sufficient monitoring of service availability. Institutions lacking these conditions may still adopt the architectural principles, but the magnitude of operational improvement must be measured locally; the 87.94% reduction observed in the supplied paired records must not be assumed to apply elsewhere.

3.11.4 Limitations and Generalization

The findings are bounded by a single-institution case and the controlled scenarios documented in the supplied evidence. Stakeholder confirmation represents five operational roles, external validation represents two experts, operational efficiency is based on 25 paired observations from five operators (five pairs each), synchronization covers 20 selected student records, and API performance is represented by a 25-sample HTML dashboard and a separate 25-row raw-log screenshot. The same student identifiers and saved repository-record outcome were used for both workflows, but the actual interaction steps differed because the To-Be process removes manual transfer. The timing workbook contains timestamps, yet no screen-level event trace independently verifies the exact start/stop event for every observation. All As-Is tasks preceded To-Be, no randomized/counterbalanced order or washout procedure was documented, and repeated observations within five operators may be correlated. The SQL file lists 20 source records and 20 after-sync repository records, but its before-state is a schematic NULL placeholder and its after-sync output omits two of the eight mapped attributes; full eight-field consistency is supported by the separate workbook audit, not independently demonstrated by the SQL output alone. The raw-log screenshot yields 184.36 ms (174–196 ms) from its visible elapsed values, whereas the HTML dashboard reports 183.12 ms (170–194 ms); the Test Plan screenshot also shows a fixed student identifier while the raw log lists 25 distinct identifiers, so the run-to-artifact correspondence remains unresolved. Native machine-readable .jmx/.jtl files remain unavailable. The uploaded Python integration script is a prototype using an in-memory mock student database and a static placeholder-token comparison, not executable evidence of a production database connection or cryptographic JWT verification. The Nginx screenshots and route-reconciliation spreadsheet do not consistently establish the internal /v1-to-/api/v1 proxy mapping. These constraints limit causal interpretation and exact independent reproduction.

The results should not be generalized as an expected 87.94% time reduction for other universities. Institutional governance maturity, data quality, API design, repository capability, operator experience, workflow design, infrastructure, network conditions, concurrency, and service availability may change both API latency and end-to-end processing time. The fixed As-Is-first order may have introduced a learning effect, and the five-operator repeated-measures structure was not analyzed with a cluster-adjusted model. The study does not establish production-scale throughput, tail latency, penetration-test assurance, long-term availability, disaster recovery, or recovery performance under an executed outage test. Future replication should counterbalance workflow order where feasible, include familiarization or washout procedures, model operator clustering, increase the number and diversity of records and

load levels, provide native JMeter .jmx/.jtl files, reconcile the JMeter dashboard with the raw execution log, verify the exact deployed Nginx upstream mapping, and publish complete anonymized before/after extracts covering all eight mapped attributes.

Taken together, the study makes three distinct but complementary contributions. First, it provides an autonomy-preserving architecture in which authoritative-source and data-ownership rules govern reuse of academic master data. Second, it provides a traceable validation chain consisting of stakeholder confirmation, external expert assessment, executed integration testing, and paired operational-time measurement. Third, it provides implementation-oriented governance controls for authentication, logging, authority allocation, rate limiting, caching or webhooks, transition planning, fallback operation, and staged deployment. Keeping governance coverage, API performance, and end-to-end workflow efficiency as separate analytical constructs prevents a fast API response or a successful technical test from being overstated as proof of institution-wide operational performance.

Table 16. Comparison with Closely Related Studies

Study	Domain / Focus	Architecture / Integration Contribution	Validation Approach	Remaining Limitation
Latifah and Samopa [11]	Higher education / management services	TOGAF ADM integrated management-information-system blueprint	Interviews, gap measurement, questionnaires, FGD, architecture validation	Primarily blueprint-level; no executed AIS–repository API or paired operational-time test
Sahara et al. [14]	Higher education / AIS-LMS	REST web-service integration and synchronization using an external-database approach	Application-level implementation and integration testing	No TOGAF-based repository integration or academic master-data ownership model
Subhiyakto et al. [13]	Integrated library catalog	RESTful integration across separate library applications	Application-level functional implementation and user evaluation	No TOGAF-based AIS-repository master-data governance
Darsono et al. [16]	Public-sector SPBE / mining	TOGAF ADM blueprint with centralized platform and API-oriented target architecture	Metric-based conceptual validation, stakeholder validation, and EA expert judgment	Efficiency values are projected/simulated; empirical post-implementation measurement remains future work
This study	Higher education / AIS-repository	TOGAF ADM with authoritative-source governance, autonomy-preserving authenticated REST API, integration layer, and audit controls	5-stakeholder confirmation; 2 external EA experts; executed functional, negative-ID, synchronization, consistency, response-time tests; 25 paired operational observations with inferential testing	Single-institution controlled evaluation; production-scale load, security assurance, long-term availability, recovery, and broader institutional generalization remain future work

To make the relationship between the research problem, TOGAF ADM decisions, implemented integration components, and evaluation evidence explicit, Table 17 consolidates the traceability chain.

Table 17. Traceability from Research Requirements to TOGAF ADM Decisions, Implementation, and Evaluation

Requirement / problem	TOGAF ADM phase	Architecture decision	Implemented component / rule	Evaluation evidence and limitation
Single authoritative source for shared academic attributes	Preliminary; Phase C	SIAKAD owns selected master attributes; repository retains publication-specific metadata	Eight-field data dictionary; read-only master fields; repository-owned publication title	20 records and 160 workbook MATCH markers; SQL snapshot lists 20 after-sync records but omits advisors_name and date_year in its displayed output; before-state is schematic, not a literal query result
Repeated manual entry between SIAKAD and repository	Phase B	To-Be workflow retrieves attributes from SIAKAD and populates repository form	Repository form invokes authenticated REST service using NIM as lookup key	25 paired operational observations; fixed As-Is-first order and possible learning effect remain
Interoperability without database consolidation	Phase A; Phase C; Phase D	Applications remain autonomous and exchange approved attributes through a service boundary	REST API, integration layer, JSON mapping, HTTPS transport	Dashboard and raw-log screenshots each show 25 successful samples but report different latency summaries; the Test Plan uses a fixed ID while the log lists 25 IDs; native .jmx/.jtl and run linkage unavailable
Controlled access to academic master data	Phase D	Token-based authentication and role-based service access	JWT Bearer policy, Nginx TLS configuration, service account ROLE_REPOSITORY_READER	HTTP 401 missing/invalid-token responses and HTTP 403 response body supplied; public /v1 endpoint supported, but Nginx excerpts do not consistently verify internal /v1-to-/api/v1 mapping
Invalid identifiers and malformed input must be handled safely	Phase C; Phase D	Validate identifier format and record existence before mapping	Input validation and structured HTTP error responses	Four HTTP 404 cases and one HTTP 400 case recorded

4. CONCLUSION

This study designs and evaluates a TOGAF ADM-based enterprise architecture for integrating an Academic Information System and an academic repository while preserving the functional autonomy of both applications. The architecture positions SIAKAD as the authoritative source for selected academic master data and provides approved attributes to the repository through the confirmed public HTTPS REST API endpoint https://api.siakad.ac.id/v1. The evaluation combines stakeholder confirmation by five operational representatives (4.74/5.00), external assessment by two EA experts (4.67/5.00), functional and security tests, synchronization of 20 records, JMeter performance evidence, and 25 paired operational observations performed by five operators. Technical evidence records 20/20 successful valid requests, four HTTP 404 responses for unregistered identifiers, one HTTP 400 response for malformed input, HTTP 401 for missing/invalid tokens, HTTP 403 for insufficient role privileges, and 20/20 records marked mapped and populated in the synchronization workbook, with 160 attribute-level MATCH markers. The SQL snapshot displays 20 after-sync records but omits two mapped attributes from its output. The JMeter dashboard reports 183.12 ms mean response time (170–194 ms), while recalculation from the separate raw-log screenshot gives 184.36 ms (174–196 ms); the Test Plan screenshot and raw-log screenshot also differ in the displayed identifier pattern, so their run-level correspondence is unresolved. Mean operational time decreased from 94.60 s to 11.408 s, a case-specific reduction of 87.94% (mean difference 83.192 s; $t(24)=74.05$; $p<0.001$; 95% CI [80.87, 85.51] s). Five operators each contributed five pairs and timestamped records are available; however, all As-

Is observations preceded To-Be, so possible learning/order effects and within-operator dependence remain. The operational improvement is therefore limited to the recorded task, operator group, sequence, and evaluated environment and should not be generalized to other universities. Future work should counterbalance order where feasible, model operator clustering, provide native JMeter exports and complete database extracts, and evaluate sustained load, security, availability, and recovery under production-like conditions.

Data availability. The anonymized 25 paired operational-time measurements and operator codes are reproduced in Appendix A; start/end timestamps for both conditions are available in the supplementary operational workbook. The supplied workbooks contain the 20-record synchronization results, eight-attribute source mapping and MATCH audit, five negative-input cases, stakeholder SV1–SV10 response matrix, and expert EV1–EV12 criterion-level scores and anonymized competence profiles. The revision package includes a normalized Postman JSON test template and environment file, screenshots of the supplied Postman scenarios and response evidence, the JMeter HTML dashboard, screenshots of the JMeter Test Plan and 25-row raw execution log, an SQL text snapshot listing 20 source and 20 repository records, OpenAPI and Nginx configuration excerpts, and HTTP 200/400/401/403/404 response examples. The uploaded Python script is a prototype based on an in-memory mock database and static placeholder-token comparison; it is not the production integration source code. The SQL snapshot is not a full database dump: its before-state is schematic and its after-sync query omits two fields from the eight-attribute mapping. The raw-log screenshot and HTML dashboard report different latency summaries, and the Test Plan screenshot cannot be conclusively linked to the raw-log run. The Nginx screenshots do not consistently verify the proposed internal route mapping. Native machine-readable JMeter .jmx/.jtl files, the original machine-readable Postman execution export, production application source code, and complete before/after database snapshots were not supplied as downloadable artifacts. Credentials, private keys, and identifiable student information are excluded or must be redacted before sharing.

REFERENCES

[1] M. A. Mohamed Hashim, I. Tlemsani, and R. Matthews, “Higher education strategy in digital transformation,” *Educ. Inf. Technol.*, vol. 27, no. 3, pp. 3171–3195, 2022, doi: 10.1007/s10639-021-10739-1.

[2] A. F. Ramadhan, F. P. Tajudeen, and N. I. Jaafar, “The Influence Factors of Data Governance Implementation: Study in Indonesian Public University,” *Procedia Comput. Sci.*, vol. 234, pp. 1204–1211, 2024, doi: 10.1016/j.procs.2024.03.116.

[3] N. S. Meutia, E. Sulistiyani, R. P. N. Budiarti, and R. Sari, “Enterprise Architecture Framework in Higher Education: Systematic Literature Review,” *Appl. Technol. Comput. Sci. J.*, vol. 5, no. 2, pp. 112–118, 2022, doi: 10.33086/atcsj.v5i2.3751.

[4] A. Lamey, H. Abdelkader, A. Keshk, and S. Eletriby, “A Realistic and Practical Guide for Creating Intelligent Integrated Solutions in Higher Education Using Enterprise Architecture,” *Sustainability*, vol. 15, no. 11, p. 8780, 2023, doi: 10.3390/su15118780.

[5] H. M. Astuti, R. P. Wibowo, and A. Herdiyanti, “Towards the National Higher Education Database in Indonesia: Challenges to Data Governance Implementation from The Perspective of a Public University,” *Procedia Comput. Sci.*, vol. 234, pp. 1322–1331, 2024, doi: 10.1016/j.procs.2024.03.130.

[6] G. N. Pramudyo and M. R. Hendrawan, “Pemilihan Perangkat Lunak Repositori Institusi Perpustakaan Perguruan Tinggi di Kota Malang,” *BACA J. Dokumentasi dan Inf.*, vol. 39, no. 2, pp. 161–177, 2018, doi: 10.14203/j.baca.v39i2.432.

[7] S. A. Salsabila, A. A. N. Fajrillah, and W. A. Nurtrisha, “Enterprise Architecture Sebagai Strategi Dalam Optimalisasi Proses dan Teknologi Menggunakan TOGAF ADM,” *JATISI (Jurnal Tek. Inform. dan Sist. Informasi)*, vol. 8, no. 4, pp. 2019–2029, 2021, doi: 10.35957/jatisi.v8i4.1129.

[8] D. Putra, D. M. K. Nugraheni, and J. E. Suseno, “Information Technology Risk-based Enterprise Architecture Design for Training Systems: Integration of COBIT 2019 and TOGAF ADM,” *J. Sains dan Teknol.*, vol. 14, no. 1, pp. 35–46, 2025, doi: 10.23887/jstundiksha.v14i1.93498.

[9] X. Sika, L. Y. Astri, D. Kisbianty, R. Setiawan, and Irawan, "Implementasi Framework TOGAF ADM Bagian Kemahasiswaan Universitas Dinamika Bangsa Jambi," *J. Ilm. Media Sisfo*, vol. 18, no. 1, pp. 1–7, 2024, doi: 10.33998/mediasisfo.2024.18.1.1453.

[10] M. R. Fanani, "Perancangan Enterprise Architecture Menggunakan TOGAF ADM untuk Integrasi Teknologi Informasi," *Kompak J. Ilm. Komputerisasi Akunt.*, vol. 17, no. 1, pp. 318–325, 2024, doi: 10.51903/kompak.v17i1.1956.

[11] Y. D. U. Latifah and F. Samopa, "Enterprise Architecture Integrated Management Information System Untuk Optimalisasi Layanan Manajemen Pendidikan Tinggi," *Kesatria J. Penerapan Sist. Inf. (Komputer dan Manajemen)*, vol. 5, no. 2, pp. 422–437, 2024, doi: 10.30645/kesatria.v5i2.352.

[12] A. N. I. Purwanto and N. Rasyid, "REST API and Message Broker RabbitMQ for Integration of College Academic Information System," *CCIT (Creative Commun. Innov. Technol. J.*, vol. 16, no. 2, pp. 177–189, 2023.

[13] E. R. Subhiyakto, Y. P. Astuti, and N. K. Ningrum, "Pengembangan Aplikasi Katalog Perpustakaan Terintegrasi Menggunakan Metode RESTful pada Perpustakaan Kota Semarang dan Perpustakaan Daerah Jawa Tengah," *J. Inform. J. Pengemb. IT*, vol. 3, no. 2, pp. 161–166, 2018, doi: 10.30591/jpit.v3i2.834.

[14] R. Sahara, S. Abdullah, M. I. Saputra, and C. R. Hassolthine, "Integration Design of Academic Information Systems and Learning Management Systems Using Web Services REST-Based External Database," *J. Ilm. FIFO*, vol. 14, no. 2, pp. 205–215, 2023, doi: 10.22441/fifo.2022.v14i2.010.

[15] L. A. D. Sari, R. Mulyana, and I. Y. Mukti, "A TOGAF 10-based enterprise architecture framework for digital transformation in SME banks," *J. Tek. Inform.*, vol. 6, no. 2, pp. 673–690, 2025, doi: 10.52436/1.jutif.2025.6.2.4329.

[16] D. Darsono, S. Febriani, and P. Rochmansyah, "Enterprise Architecture Design of SPBE Using TOGAF ADM for Improving Operational Efficiency: A Case Study of the Mining Sector at the West Java ESDM Agency," *JEEICT*, vol. 8, no. 1, pp. 33–39, 2026, doi: 10.20961/jeeict.8.1.111052.

Vol. 11, No. 2, Oktober 2026, Page. 621-643
https://journal.uin-alauddin.ac.id/index.php/instek
p-ISSN: 2541-1179
e-ISSN: 2581-1711
JURNAL INSTEK
INFORMATIKA SAINS DAN TEKNOLOGI

APPENDIX A. RAW PAIRED OPERATIONAL-TIME OBSERVATIONS

The table reproduces the 25 paired values from the operational workbook using anonymized sample and operator codes. Difference is calculated as As-Is minus To-Be. The full supplementary workbook records start and end timestamps for both conditions. The fixed execution order was As-Is first and To-Be second on 14 August 2026.

Sample	Operator	As-Is (s)	To-Be (s)	Difference (s)
Obs-01	O1	95.0	11.500	83.500
Obs-02	O2	92.0	11.000	81.000
Obs-03	O3	104.0	12.500	91.500
Obs-04	O4	88.0	10.500	77.500
Obs-05	O5	93.0	11.200	81.800
Obs-06	O1	110.0	14.000	96.000
Obs-07	O2	85.0	10.000	75.000
Obs-08	O3	97.0	11.800	85.200
Obs-09	O4	91.0	11.000	80.000
Obs-10	O5	102.0	12.200	89.800
Obs-11	O1	89.0	10.800	78.200
Obs-12	O2	96.0	11.500	84.500
Obs-13	O3	94.0	11.300	82.700
Obs-14	O4	83.0	9.800	73.200
Obs-15	O5	105.0	13.000	92.000
Obs-16	O1	90.0	10.700	79.300
Obs-17	O2	98.0	12.000	86.000
Obs-18	O3	92.0	11.100	80.900
Obs-19	O4	101.0	12.400	88.600
Obs-20	O5	87.0	10.200	76.800
Obs-21	O1	95.0	11.400	83.600
Obs-22	O2	99.0	12.100	86.900
Obs-23	O3	93.0	11.000	82.000
Obs-24	O4	88.0	10.500	77.500
Obs-25	O5	98.0	11.700	86.300