JURNAL EMACS
[Engineering, MAthematics and Computer Science] Vol.8 No.1 January 2026: 105-112 DOI: 10.21512/emacsjournal.v8i1.15355
e-ISSN: 2686-2573

Design and Implementation of Web Service-Based ETL Middleware for PDDIKTI Reporting at XYZ University

Harkat Christian Zamasi

Information Systems Department
School of Information Systems
Bina Nusantara University
Jakarta, Indonesia 11480
harkat@binus.ac.id

Correspondence: harkat@binus.ac.id

Abstract – *The reporting of academic data to the Higher Education Database (PDDIKTI) is an obligation for all universities in Indonesia, which must be carried out properly and in a timely manner according to government regulations. However, reporting is still largely done manually via the Neo Feeder application, resulting in inefficiencies, inconsistencies in the data, and possible delays in reporting, especially in universities with large amounts of academic data. Web service facilities are available, but they are not yet used in an optimal way. Universities' integration of internal academic information systems with PDDIKTI is hindered by the heterogeneity of data structures, business rules, and technology platforms, and the unavailability of a more generally replicable middleware model. In this study, we design and implement a web service-based ETL middleware architecture using Neo Feeder as a solution for automating data reporting. The research employed an applied research method with an information systems engineering approach, consisting of requirements analysis, system architecture design, prototype implementation, and system evaluation using a case study at XYZ University. The proposed system successfully automated the transfer of academic data records through iterative batch processing, reducing repetitive manual reporting activities and improving reporting efficiency. The proposed model can also be replicated by other universities.*

Keywords: *PDDIKTI; Neo Feeder; web services; middleware ETL; academic information system; data integration*

I. INTRODUCTION

Pangkalan Data Pendidikan Tinggi (PDDIKTI) is a national database that serves as the primary source of information on the implementation of higher education in Indonesia. Based on Article 56 paragraphs 1 and 2 of Law No. 12 of 2012 on Higher Education in Indonesia, PDDIKTI plays a strategic role in supporting the one data policy for Higher Education and plays a vital role in the accreditation process, quality evaluation, educational planning, and decision-making by the government and other stakeholders. In accordance with Regulation of the Minister of Research, Technology, and Higher Education Number 61 of 2016, all state and private universities are required to report their academic data completely, accurately, and timely into the PDDIKTI system for each reporting period. The timeliness and quality of reported data are crucial factors because data is used as an official reference in various strategic processes, such as study program and university accreditation, lecturer workload (BKD), academic position level (JJA), and government guidance and supervision policies (Cunha et al., 2019; Falani et al., 2024; Ngatmari et al., 2020).

To realize one national data, the government created an integrated database system, known as PDDIKTI. This initiative is part of one data Indonesia, regulated by Presidential Regulation No. 39 of 2019, which aims to provide credible, accountable, up-to-date, and integrated data to support transparency and public data access (Ngatmari et al., 2020; Saepudin et al., 2025). According to the PDDIKTI website, there are 4,416 state and private universities across the archipelago. Integrating this data is a significant challenge for both the government and universities as educational units. This university data integration involves the consolidation of data from each university into a national database data warehouse system (Islami, 2021; Perwira & Santosa, 2017).

PDDIKTI data reporting is conducted through the Neo Feeder application, an official client application provided by the government and installed in each university environment. Neo Feeder provides facilities for inputting, validating, and synchronizing academic data from universities to the national database (Brawijaya et al., 2023; Falani et al., 2024). Neo Feeder has provided web services, in order to universities to send data, integrate with their internal systems with PDDIKTI through an automated Extract, Transform, and Load (ETL).

However, in many universities, particularly those with large-scale and highly complex academic data environments, the data reporting process through the Neo Feeder interface is still performed manually or semi-manually. This approach raises some challenges such as time-consuming due to repetitive manual data entry activities, high dependency on operators, and higher risk of input errors and data inconsistency between the university internal Academic Information System and PDDIKTI. The challenges are greater in universities with many students and wide academic activities, where the amount of academic data to be reported each semester is large and involves many academic entities (Perwira & Santosa, 2017; Rahim et al., 2025). Repetitive activities, particularly manual data entry, result in inefficiency in the reporting process, which can reduce data quality control and could even lead to reporting delays.

Universities have historically developed their own Academic Information Systems to support their internal operational needs. These systems maintain data about students, faculty, courses, grades, and other academic activities. However, the variety of platforms, database structures and internal business rules often present challenges when integrating this data with external, national scale systems (Toni & Hadi, 2023). Integration of academic information systems is a strategic need so that operational data can be utilized not only for internal purposes, but also for reporting and compliance with external regulations, making an important contribution to increasing the efficiency of academic services, data transparency, and user satisfaction (Hakim et al., 2025), and facilitate the management of large volumes of information quickly and accurately (Siregar & Situmeang, 2022). In the context of higher education data reporting, good integration enables academic data to be managed consistently and accurately and improves the quality and reliability of data reported to the government.

Middleware is a general term for software that is used as an intermediary or communication medium between separate applications in a heterogeneous system environment (Alamsyah et al., 2019; Asif & Webb, 2015; Connolly & Begg, 2015; Lesmono et al., 2022), functions as an intermediary layer between the source system and the destination system in a distributed system architecture, tasked with managing communication, data transformation, validation, and controlling the integration process flow between systems using web services technology (Schantz & Schmidt, 2007), without having to understand the internal architecture of each service (Samuel, 2014; Zhuang et al., 2022). Middleware can be beneficial as it allows flexibility, scalability and ease of maintenance of the system (Schlesinger et al., 2005). Hence, the middleware application is an intermediate layer that facilitates data interchange and integration between the internal systems of the university and Neo Feeder and overcomes the incompatible underlying technology platforms. The middleware should enable automated system-to-system data reporting processes, reducing the reliance on manual data entry by operators.

ETL (Extract, Transform, and Load) is a common data integration approach, especially when data comes from different operational sources and needs to be adapted to a specific target system. The ETL process is a process of extracting data from the source database, transforming data according to applicable formats and rules, and loading data into the target system (Wijaya & Pudjoatmodjo, 2016; Yulianto, 2019). The ETL process plays an important role because the data structure in the higher education academic information system is often not entirely the same as the Neo Feeder data structure (Rahim et al., 2025), and compliance with the validation rules set by PDDIKTI.

Data reporting to PDDIKTI encompasses various academic entities representing higher education activities, each governed by specific reporting rules, deadlines, and government regulations that universities must comply with. The reported entities include: 1) study plan data, such as classes, lecture schedules, grades, and course–student relationships, which must be reported no later than two months after the beginning of the semester; 2) lecturer teaching data, including courses and class schedules taught by lecturers, which are also used for lecturer workload reporting, academic promotion, certification, and performance evaluation; 3) student grade data, reported after the completion of lectures and used to generate academic summaries such as credit accumulation and GPA; 4) new student profile data, which must be reported before the reporting period ends; 5) course and curriculum data representing the academic structure of each study program; 6) student activity data, including theses, enrichment programs, Independent Campus activities, grade conversions, and academic supervision records; and 7) student status updates, including graduation, withdrawal, and dropout information, to ensure accurate academic status records. Furthermore, the PDDIKTI reporting

process also involves several other main and supporting academic data entities .

Several previous studies have discussed the integration of academic systems with PDDIKTI, among others the development of data warehouses and database synchronization applications to improve the efficiency of academic data reporting. Most of these studies, however, mainly focus on data synchronization or particular reporting functions and have not fully addressed the design of an integrated ETL middleware architecture that can be replicated by other universities and supports automated system-to-system data transfer.

Based on these problems, this research proposes the design of ETL middleware based on Neo Feeder web services to automate the PDDIKTI data reporting process. The case study is XYZ University in Jakarta, a large private university with more than 55,000 active students at various levels and study programs. The proposed middleware is a middle layer that connects the operational academic database of the university to the PDDIKTI system with a standard ETL mechanism and facilitates the automatic data transfer through batch processing approach.

II. METHODS

This research applies an applied research approach and an information systems engineering method for designing and implementing ETL middleware based on Neo Feeder web services to automate PDDIKTI data reporting from the university's academic information system.

The first stage of this research was carried out by analyzing needs and problems through the study of PDDIKTI regulatory documents, Neo Feeder documentation, and observations of the data reporting process that is currently in progress. This stage aimed to identify constraints in the manual input process and data structure inconsistencies between the internal academic system and PDDIKTI. Second, the ETL middleware architecture was designed to serve as an integration layer between XYZ University's academic database and Neo Feeder's web services. The architecture was designed to support data extraction, data transformation according to the PDDIKTI schema, and automatic data loading through web services. Third, the middleware was implemented in the form of a web-based application that supports a scheduled and controlled ETL process, using batch processing for gradual data transfer. Fourth, the system was tested functionally and integratively to ensure that academic data could be reported correctly and consistently.

Evaluation was conducted by comparing the reporting process before and after the implementation of the ETL middleware in terms of reporting efficiency. The evaluation measured the percentage of efficiency improvement by comparing the time required for manual data entry and automated batch-processing–based transfer processes.

III. RESULTS AND DISCUSSION

3.1 Architecture Model

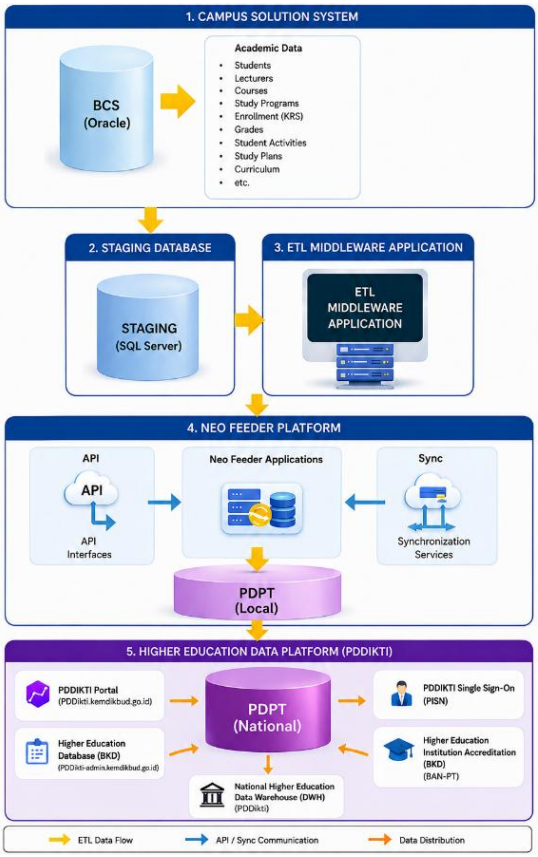

Figure 1. Architecture Model ETL Middleware for PDDIKTI Reporting

Figure 1 depicts the proposed architectural model consists of three main layers: the data source layer, the ETL middleware layer, and the PDDIKTI target layer. The data source layer represents the university's internal academic database, which stores operational academic data. The ETL middleware layer is responsible for extracting, transforming, validating, and transferring data, while also managing communication with Neo Feeder through web services. The target layer represents the PDDIKTI system as the national higher education database used for academic data reporting and synchronization.

3.2 ETL Application Middleware

In addition to the architectural model concept, this study also developed a middleware prototype to prove that the proposed model can be practically implemented and works according to PDDIKTI data reporting requirements. The ETL middleware prototype is developed using PHP programming language. The ETL middleware uses an ODBC driver to connect to the staging database.

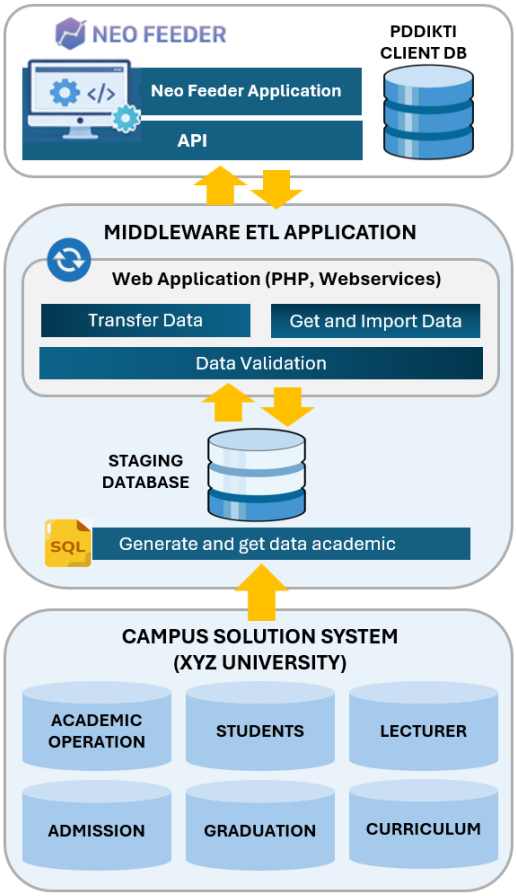

Figure 2. ETL Application Middleware Architecture

Figure 2 illustrates the detailed architecture of the ETL middleware application used to manage the extraction, transformation, validation and transfer of academic data between the university academic system and Neo Feeder via web service communication.

During the extraction phase, the middleware extracts the academic data from the staging database. The staging database is an intermediate layer between the academic system used by the university and the PDDIKTI system. The staging database allows for data processing, validation and transformation to take place without directly impacting on the operational academic database.

The transformation phase is where internal academic data is mapped and transformed into a structure that is compatible with the Neo Feeder schema. This includes changes to data types, field structures and validation requirements. This process also includes data cleansing, standardization of the format and handling of incomplete or null values to ensure compliance with the PDDIKTI validation rules.

During loading phase, the middleware transfers data to Neo Feeder using web service communication with methods based on the data operation required. In case of failure of transfer process system gets a response with transfer status, generated record identifier and validation error messages.

3.3 Data Transfer Mechanism

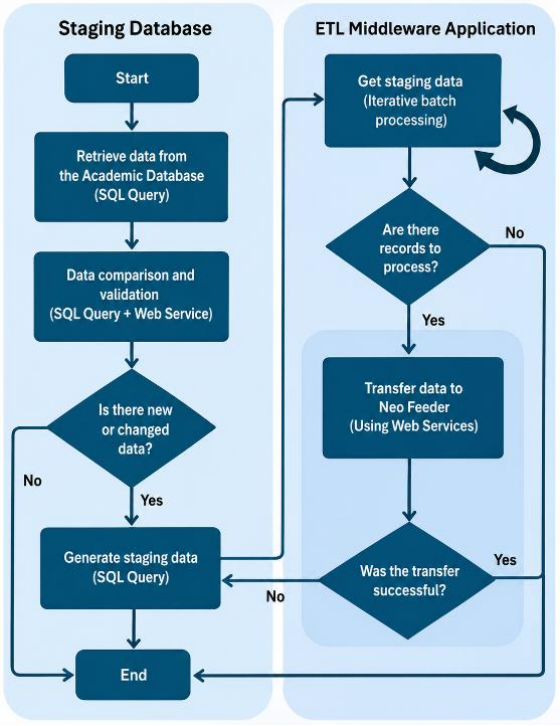

Figure 3. Batch Processing-Based Academic Data ETL Process Flow

Figure 3 shows the batch processing-based ETL workflow used by the middleware application to automate the academic data reporting to PDDIKTI. The process begins by extracting academic data from the university academic database using SQL queries. The retrieved data is then compared and validated with existing data in PDDIKTI through web service communication to find new or updated records. . If changes are detected the middleware creates staging data in the staging database as an intermediate data store.

Once the staging data is generated, the middleware goes through batch processing. For each transfer cycle, a limited number of records are fetched from the staging database. The mechanism is intended to handle large amounts of academic data in a controlled and stable way. The selected records

are then fed to Neo Feeder by web service methods as per the required data operation.

The transfer result of each batch is checked by the middleware during the transfer process. If the transfer is successful, the process continues with the next batch of records until there is no more data to process. If the transfer fails, the records are kept in the staging database for monitoring, validation, and retry processing. This workflow allows for automated synchronization, reduces manual and repetitive input activities, and improves the reliability and consistency of PDDIKTI data reporting.

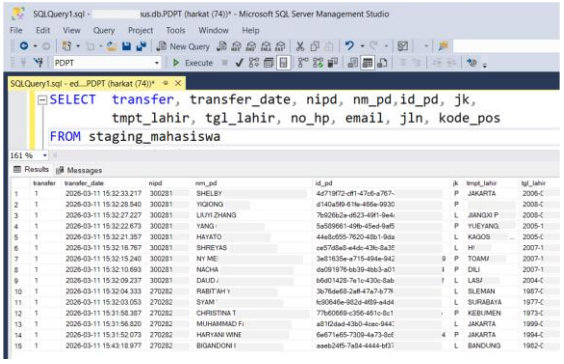

Figure 4. Student Data Staging Table with Transfer Status to Neo Feeder

The student staging table is shown in Figure 4 and is used by the ETL middleware application to store the extracted and transformed data before it is sent to the Neo Feeder system via web service. Transfer column is a data transfer status indicator. 0 means data ready to be sent, 1 means data successfully transferred, and 2 means data that failed or encountered an error. This status marking mechanism supports batch processing, monitoring, and auditing of data transfers. Furthermore, this status is used as batch process control, a data transfer audit trail, and a retry mechanism for data that has not been successfully sent.

In addition to sending data to PDDIKTI, the prototype also supports importing data from Neo Feeder to a staging database. This process involves batch processing data through PDDIKTI's web services, with a specific number of records. This approach is used primarily for monitoring and data reconciliation purposes. Second, the staging data is used for data validation to ensure successful delivery. Validation is then performed by comparing the data with SQL.

Figure 5. Implementation of Calling the Web Service to Insert Student Biodata

The data that has undergone the cleansing and validation process is then read from the staging table and prepared for transfer to the Neo Feeder system. This process is performed by executing a limited query using a Top-N approach (e.g., 10 records) in each processing cycle. The retrieved data is then processed by a PHP script and sent to Neo Feeder via a web service. To support incremental batch processing, PHP web pages automatically refresh using the HTML meta refresh feature as long as there is pending data. The refresh interval is determined by the time parameter (t seconds) in the content attribute of the <meta http-equiv="refresh"> element.

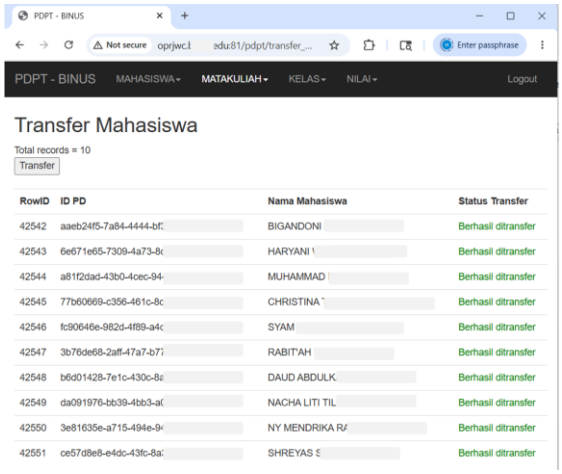

Figure 6. Iterative Batch Data Transfer Process to Neo Feeder

The Top-N approach is implemented because transferring large amounts of data via web services has performance and reliability limitations. Hence, data transfer is performed in small chunks per cycle, e.g. 10 records per process, in order to keep the system stable, reduce potential failures and facilitate the monitorization and error handling.

3.4 Evaluation

Based on operational observations and human computer interaction estimates, the manual entry of a single student profile record through the application interface may take approximately 45 - 90 seconds per record, even when the data is available in digital form and entered using copy and paste activities. The required time includes field navigation, validation checking and data submission processes, while the batch-processing transfer mechanism requires approximately 3 second per record.

To give some sense of the efficiency difference between the two approaches, let's assume that manual entry takes about a minute a record. There are 1,000 records in the data set. The manual reporting process would take about 1,000 minutes whereas the batch-processing transfer approach would take about 50 minutes. The efficiency gain was calculated by using the following formula:

$$E\ (\%) = \frac{M-T}{M} \times 100\% \qquad (1)$$

E = Efficiency Improvement (%)
M = Manual Entry Time
T = Batch Processing Transfer Time

By this calculation, the implementation of batch processing-based ETL middleware has an efficiency improvement of about 95 percent over the traditional manual reporting process.

3.5 Discussion

The results of the implementation show that the proposed middleware architecture can improve the efficiency and reliability of the PDDIKTI reporting process by reducing the repetition of manual input activities and allowing the automated batch-based data transfer between the academic system and Neo Feeder. Furthermore, the utilization of a staging database and transfer status indicators facilitates monitoring, error tracking and retry mechanisms for failed records, thereby enhancing data consistency and reporting control. The prototype implementation shows that the developed ETL middleware architecture can be successfully used for automating PDDIKTI reporting.

Additionally, the middleware's ability to manage large volumes of academic data in a controlled and staged manner is demonstrated by the batch-processing method and the record number limitation mechanism. This approach is especially relevant for large universities like XYZ University that manage tens of thousands of active students.

The paper proposes a generic and flexible ETL middleware architecture. The prototype was implemented with a case study of XYZ University but the proposed architecture, ETL workflow and integration mechanisms can be replicated by other universities with modifications made to their database structures and data mapping needs.

IV. CONCLUSION

The proposed ETL middleware successfully automated the transfer of academic data records through iterative batch-processing-based system-to-system communication, so that repetitive manual reporting activities could be reduced and the efficiency and reliability of PDDIKTI data reporting was improved. The introduction of a staging database, transfer status monitoring, and retry mechanisms also help in better data validation, error handling, and reporting control, resulting in a more structured, scalable, and integrated reporting process. Moreover, the proposed architecture is generic and flexible, which can be replicated by other universities with modifications to their respective academic data structures and operational requirements.

Future research may focus on developing cloud-based integration models, enhanced security and data governance frameworks, as well as intelligent monitoring and automated validation techniques to further improve the reliability and efficiency of PDDIKTI data reporting across higher education institutions despite these promising results.

REFERENCES

Alamsyah, F., Witanti, W., & Hadiana, A. I. (2019). (*Design of an Integrated Sales Information System Using Web Services at CV. Sejahtera*). Perancangan Sistem Informasi Penjualan Terintegrasi Menggunakan Web Service Di CV. Sejahtera. *Prosiding Sains Nasional Dan Teknologi*, *1*(1). https://doi.org/10.36499/psnst.v1i1.2873

Asif, S., & Webb, P. (2015). Software System Integration Middleware an Overview. *International Journal of Computer Applications*, *121*(5), 27–29. https://doi.org/10.5120/21538-4547

Brawijaya, H., Samudi, S., & Widodo, S. (2023). Implementation of Pddikti Neo Feeder Web Service in Recording of Independent Campus Activities. *Jurnal Riset Informatika*, *5*(2), 203–210. https://doi.org/10.34288/jri.v5i2.500

Connolly, T., & Begg, C. (2015). *Database Systems. A Practical Approach to Design, Implementation, and Management.* (SIXth edit). https://doi.org/10.1007/978-1-4842-1191-5

Cunha, T. E. B. da, Tuba Helan, Y. G., & Tadeus, D. W. (2019). (*The Implementation of the Feeder Application in Relation to the Protection of Lecturers' and Students' Personal Data: A Review of Law No. 12 of 2012 on Higher Education*). Implementasi Aplikasi Feeder Dihubungkan Dengan Perlindungan Data Pribadi Dosen Dan Mahasiswa Ditinjau Dari Undang-Undang Nomor 12 Tahun 2012 Tentang Pendidikan Tinggi. *Jurnal Akrab Juara*, *4*(3), 21–30. http://www.akrabjuara.com/index.php/akrabjuara/article/view/707

Falani, A. Z., Kamisutara, M., & Artaya, I. P. (2024). (*The Use of Web Services and Microsoft Excel for Reporting PD DIKTI Neo Feeder Data at STIKES KH Kediri*). Pemanfaatan Web Services Dan Microsoft Excel Untuk Pelaporan Data Neo Feeder PD DIKTI di STIKES KH Kediri. *Asthadarma : Jurnal Pengabdian Kepada Masyarakat*, *5*(2), 73–81. https://doi.org/10.55173/Asthadarma.v5i2.27

Hakim, H. Q. I., Bilqis, S. S., Ramadhani, R., & Vindua, R. (2025). (*Literature Review: The Effectiveness of Web-Based Academic Information Systems in Higher Education Institutions*). Tinjauan Literatur: Efektivitas Sistem Informasi Akademik Berbasis Web Di Perguruan Tinggi. *JUTECH: Journal Education and Technology*, *6*(1), 74–86. https://doi.org/10.31932/jutech.v6i1.4845

Islami, M. J. (2021). Implementasi Satu Data Indonesia: Tantangan dan Critical Success Factors (CSFs). Jurnal Komunika: Jurnal Komunikasi, Media Dan Informatika, 10(1), 13-23. https://doi.org/10.31504/komunika.v10i1.3750

Lesmono, A., Ghozali, K., & Indrawanti, A. S. (2022). (*Data Integration in Desktop Applications*). Integrasi Data pada Aplikasi Desktop. *Jurnal Teknik ITS*, *11*(2). https://doi.org/10.12962/j23373539.v11i2.85268

Ngatmari, N., Musthafa, M. B., Rahmad, C., Asmara, R. A., & Rahutomo, F. (2020). (*The Use of PDDIKTI Data to Support Management Decision-Making in Higher Education Institutions*). Pemanfaatan Data PDDIKTI sebagai Pendukung Keputusan Manajemen Perguruan Tinggi. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, *7*(3), 555–564. https://doi.org/10.25126/jtiik.2020722585

Perwira, R., & Santosa, B. (2017). (*Implementation of Web Services in the Integration of Academic Data with the Dikti Database Replica*). Implementasi Web Service pada Integrasi Data Akademik dengan Replika Pangkalan Data Dikti. *Telematika*, *14*(01). https://doi.org/10.31315/telematika.v14i01.1962

Rahim, A., Wardani, M., Alam Jusia, P., & Siswanto, A. (2025). (*Implementation and Development of the PDDikti Feeder Academic Dashboard Using a Data Mart: A Case Study of Universitas Dinamika Bangsa*). Implementasi Dan Pengembangan Dashboard Akademik Feeder PDDikti Dengan Data Mart: Studi Kasus Universitas Dinamika Bangsa. *Jurnal Informatika Dan Rekayasa Komputer(JAKAKOM)*, *5*(1), 1336–1345. https://doi.org/10.33998/jakakom.2025.5.1.2184

Saepudin, E. A., Solehatunnisa, F., Lestari, N. A., Gunawan, R. A. V., & Fadilah, S. (2025). (The Implementation of Satu Data Indonesia in Supporting Transparency and Access to Public Data). Implementasi Satu Data Indonesia dalam Mendukung Transparansi dan Akses Data Publik. *Jejak Digital: Jurnal Ilmiah Multidisiplin*, *1*(3), 442–452. https://doi.org/10.63822/pydk5v51

Samuel, J. (2014). *Towards a Data Warehouse Fed with Web Services* (pp. 874–884). https://doi.org/10.1007/978-3-319-07443-6_61

Schantz, R. E., & Schmidt, D. C. (2007). Middleware for Distributed Systems. In *Wiley Encyclopedia of Computer Science and Engineering*. Wiley. https://doi.org/10.1002/9780470050118.ecse241

Schlesinger, L., Irmert, F., & Lehner, W. (2005). Supporting the ETL-process by Web Service technologies. *International Journal of Web and Grid Services*, *1*(1), 31. https://doi.org/10.1504/IJWGS.2005.007547

Siregar, B., & Situmeang, M. (2022). (*The Use of SIAKAD to Support the Delivery of Education and Its Benefits for Institutions and Students*). Pemanfaatan SIAKAD dalam Menunjang Pelaksanaan Pendidikan serta Manfaatnya bagi Institusi dan Mahasiswa. *All Fields of Science Journal Liaison Academia and Sosiety*, *2*(4), 210–216. https://doi.org/10.58939/afosj-las.v2i4.485

Toni, M., & Hadi, A. (2023). (*Development of the Academic Information System at LP3I Polytechnic, Padang Campus, Using the Laravel Framework*). Pengembangan Sistem Informasi Akademik Politeknik LP3I Kampus Padang Menggunakan Framework Laravel. *Jurnal Sains Dan Teknologi Informatika*, *1*(2), 73–79. https://doi.org/10.38204/jsti.v1i2.1582

Wijaya, R., & Pudjoatmodjo, B. (2016). (*The Application of Extraction-Transformation-Loading (ETL) in Data Warehousing - Case Study: Ministry of Agriculture*). Penerapan Extraction-Transformation-Loading (ETL) Dalam Data Warehouse (Studi Kasus : Departemen Pertanian). *Jurnal Nasional Pendidikan Teknik Informatika (JANAPATI)*, *5*(2), 61. https://doi.org/10.23887/janapati.v5i2.9855

Yulianto, A. A. (2019). Extract Transform Load (ETL) Process in Distributed Database Academic Data Warehouse. *APTIKOM Journal on Computer Science and Information Technologies*, *4*(2), 61–68. https://doi.org/10.11591/APTIKOM.J.CSIT.36

Zhuang, L., Chen, Y., Cai, Q., Wang, P., & Xu, Z. (2022). Research on Data Interaction and System Integration of Heterogeneous Systems Based on Web Services. *2022 6th International Conference on Wireless Communications and Applications (ICWCAPP)*, 59–62. https://doi.org/10.1109/ICWCAPP57292.2022.00022