Journal of Inkofar (Industrial Engineering, Computer Science and Pharmacy)
Vol.10, No.1, 2026
Puspabhuana & Triyana,2026

DESIGN OF A SERVERLESS LECTURER ACADEMIC PORTFOLIO PLATFORM USING JAMSTACK AND SUPABASE

Adam Puspabhuana, Data Science, Computer Science, Jakarta, adampuspa.bhuana@cakrawala.ac.id[1*]
Yudi Triyana, Information System, Computer Science, Jakarta, yuditriyana@cakrawala.ac.id[2]

Corrsepondensi Authors : adampuspa.bhuana@cakrawala.ac.id

Abstract

Background : Digitalizing academic portfolios is an urgent need for lecturers in Indonesia to enhance their scientific visibility and meet accreditation requirements. However, the currently available Content Management System (CMS) platforms are generic and not designed to accommodate the specific needs of academic portfolios for lecturers.

Purpose : This research aims to design and implement a serverless CMS platform tailored for faculty academic portfolios, using a zero-configuration deployment approach

Methodology : The research adopts the Design Science Research (DSR) framework comprising six stages. Evaluation was conducted through black-box testing with 20 scenarios derived from the functional requirements specification, comparative analysis using five evaluation criteria (academic features, deployment complexity, infrastructure requirements, security mechanisms, and operational cost), and heuristic evaluation involving five evaluators consisting of usability experts and information systems lecturers based on Nielsen's ten usability principles.

Findings : This platform was successfully developed using the Jamstack architecture, with Supabase as the Backend-as-a-Service (BaaS). All 20 functional test scenarios were validated with a 100% success rate. Heuristic evaluation yielded an average score of 4.1 out of 5

Implications : The research results contribute to the development of a serverless academic web application architecture and offer practical solutions for digitalizing faculty portfolios in Indonesia

Originality : Beyond implementing a serverless portfolio platform, this research contributes a reusable architectural model for academic CMS development that integrates Jamstack, Backend-as-a-Service, zero-configuration deployment, and multi-layer security architecture..

Keywords *: Content Management System; Jamstack; Backend-as-a-Service; Academic Portfolio; Design Science Research.*

INTRODUCTION

Digital transformation in the higher education sector has driven fundamental changes in how academic institutions manage and present information. Lecturers, as the forefront of knowledge production, are required not only to produce high-quality scientific works but also to build academic visibility through a structured and professional digital presence (Sulistyawati & Santoso, 2022). A digital academic portfolio is a strategic instrument that summarizes all dimensions of a lecturer's contributions, including professional profiles, educational backgrounds, lists of scientific publications, innovation products, community service activities, and courses taught (Rahardja et al., 2023). The availability of a comprehensive digital portfolio supports academic personal branding, the accreditation process of research programs, and cross-institutional research collaboration (Aini et al., 2022).

Although the urgency of digitalizing portfolios has been recognized, its implementation faces several significant obstacles. First, developing a website requires proficiency in programming languages and server configuration, which are skills outside the domain of non-IT faculty members

(Handayani et al., 2022). Second, generic CMS platforms like WordPress require complex customization to accommodate specific academic features such as DOI metadata recording and semester data management (Wahyudi & Pratama, 2023). Third, traditional server architecture requires ongoing maintenance and operational costs that are not trivial for individual faculty members (Pramudyo et al., 2022).

A review of previous research shows that most faculty portfolio systems adopt a monolithic architecture that requires a dedicated server (Setiawan et al., 2022), use generic CMS with limited customization (Nugroho & Wibowo, 2023), or require DevOps expertise for deployment (Kurniawan et al., 2023). This condition indicates a research gap: the absence of a CMS that simultaneously (1) is specifically designed for academic portfolios, (2) does not require a traditional server, and (3) can be deployed without programming skills.

To bridge this gap, this research proposes a serverless CMS platform specifically designed for academic faculty portfolios. The CMS is built using a Jamstack architecture combined with Supabase as a Backend-as-a-Service (BaaS). The main uniqueness of the CMS includes a zero-configuration deployment approach via a browser wizard and a multi-layer security architecture that integrates Row Level Security, bcrypt hashing, secret key protection, and role-based access control. The research adopts a Design Science Research (DSR) methodology with contributions: (1) a purpose-built serverless CMS architecture, (2) a zero-configuration deployment framework, (3) a multi-layer security model, and (4) empirical validation through functional testing and comparative analysis.

To further clarify the identified research gap, Table 1 summarizes representative studies on lecturer portfolio systems and related web architectures. The comparison highlights that existing solutions generally rely on monolithic architectures, generic CMS platforms, or complex deployment processes. To the best of our knowledge, no previous research simultaneously integrates a serverless Jamstack architecture, Backend-as-a-Service, zero-configuration deployment, and multi-layer security within a purpose-built academic portfolio platform.

Table 1. Comparison of Previous Research

No	Researcher (Year)	Main Findings	Gaps/Limitations
1	Setiawan et al. (2022)	Laravel Lecturer Portfolio	Dedicated server
2	Nugroho & Wibowo (2023)	WordPress for lecturers	Limited academic features
3	Rahardja et al. (2023)	Digital React Portfolio	Complex deployment
4	Biilmann & Hawksworth (2022)	Jamstack superior	Not an academic domain
5	Chandra & Kumar (2022)	BaaS accelerates development	Without academic cases
6	Handayani et al. (2022)	Digitalization of lecturer data	Need DevOps

LITERATURE REVIEW

Content Management System (CMS) is software that enables users to create, manage, and modify digital content without requiring advanced programming skills (Barker, 2021). Based on its architecture, CMS platforms are generally classified into traditional CMS (monolithic), headless CMS (API-first), and static site CMS (Jacobsen & Schlenker, 2023). Traditional CMS integrates the presentation layer, business logic, and database into a single application, whereas headless and static CMS architectures separate content management from content presentation, providing greater flexibility, scalability, and maintainability. In the context of lecturer academic portfolios, an effective CMS should not only simplify content management but also provide domain-specific modules, including publication management, research activities, community service, teaching portfolios,

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

intellectual property, and professional achievements, while remaining accessible to users with limited technical expertise.

Jamstack is a modern web development architecture that separates the presentation layer from backend services through reusable APIs and static asset delivery (Biilmann & Hawksworth, 2022). Recent empirical studies have demonstrated that Jamstack architecture significantly improves deployment efficiency, scalability, and website performance compared with traditional server-rendered architectures. Siriwardena and Dias (2023) reported that the decoupled frontend architecture reduces the attack surface while improving maintainability through API-driven communication. Similarly, Wen et al. (2025) observed that serverless environments provide elastic resource allocation and competitive response times under dynamic workloads, although cold-start latency may still occur in infrequently accessed services. These findings indicate that Jamstack is particularly suitable for academic web applications requiring low operational overhead and simplified deployment.

Compared with traditional monolithic web architectures, Jamstack minimizes server-side processing, reduces the attack surface, improves loading performance through distributed Content Delivery Networks (CDNs), and simplifies deployment using static hosting services. Backend-as-a-Service (BaaS) complements this architecture by providing managed backend services, including databases, authentication, object storage, and serverless functions, thereby eliminating the need for conventional server administration (Chandra & Kumar, 2022). Supabase, as an open-source BaaS platform, provides PostgreSQL databases, Row Level Security (RLS), authentication services, object storage, automatic RESTful APIs through PostgREST, and Remote Procedure Call (RPC) functions that significantly simplify the development of cloud-native applications.

Recent empirical studies provide stronger evidence supporting the adoption of serverless and cloud-native architectures for modern web applications. Wen et al. (2025) proposed the SCOPE framework for performance testing of serverless computing and demonstrated that serverless platforms achieve competitive response times while efficiently utilizing computational resources under varying workloads. Their findings also highlight the importance of systematic benchmarking to evaluate latency, scalability, and runtime performance in serverless environments. Complementing this work, Wen, Chen, Sarro, and Wang (2025) reported that although serverless architectures provide excellent elasticity and operational efficiency, performance variance and cold-start latency remain important considerations when designing production-grade applications. Furthermore, Bodner et al. (2025) empirically showed that serverless cloud infrastructures substantially reduce infrastructure management complexity while maintaining scalable execution for large-scale workloads, although deployment cost and workload characteristics should still be carefully considered. Collectively, these empirical findings provide strong justification for adopting a Jamstack and Backend-as-a-Service architecture in this research because the proposed lecturer academic portfolio platform prioritizes simplified deployment, automatic scalability, maintainability, and reduced operational overhead.

Recent benchmarking studies further indicate that managed cloud database platforms based on PostgreSQL provide reliable transaction processing, simplified operational maintenance, and horizontal scalability compared with self-managed database servers (Zbořil & Svatá, 2025). Since Supabase is built upon PostgreSQL, it inherits these database capabilities while extending them through integrated authentication, Row Level Security, object storage, automatic API generation, and serverless execution. These integrated services significantly reduce backend development complexity while maintaining security and database reliability. In addition, comparative evaluations of modern web development technologies demonstrate that JavaScript-based cloud-native architectures consistently outperform conventional server-rendered applications in deployment flexibility and development efficiency, particularly when combined with API-driven architectures (Swedheetha et al., 2024). These findings further support the selection of Jamstack and Supabase as the technological foundation of the proposed platform.

Recent studies on academic information systems indicate that cloud-native platforms simplify long-term maintenance while improving institutional scalability. Kurniawan et al. (2023) demonstrated that microservice-based academic systems improve modularity but require DevOps expertise for deployment. Similarly, Rahardja et al. (2023) reported that digital academic portfolios increase lecturers' research visibility; however, deployment complexity remains a significant barrier for non-technical users. These findings suggest that existing academic platforms still require considerable

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

technical expertise, thereby motivating the development of a serverless academic portfolio platform with zero-configuration deployment.

Although previous empirical studies have confirmed the technical advantages of serverless and cloud-native architectures, their implementation in lecturer academic portfolio systems remains relatively limited. Most existing studies evaluate infrastructure performance, cloud deployment strategies, or database technologies independently, rather than integrating these technologies into a domain-specific academic information system. Consequently, understanding the functional characteristics of digital academic portfolios becomes essential before proposing an architectural solution tailored to higher education institutions.

The digital academic portfolio is a structured collection of lecturers' academic achievements in digital form that enables continuous updates and integration with scholarly ecosystems such as ORCID and Google Scholar (Barrett, 2021). A comprehensive digital academic portfolio generally consists of professional profiles, educational background, employment history, scientific publications, research projects, innovation products, intellectual property, teaching responsibilities, community service activities, awards, certifications, and professional contact information (Aini et al., 2022). Within Indonesian higher education institutions, digital academic portfolios have become increasingly important because accreditation agencies require structured documentation of lecturers' academic performance to support institutional quality assurance and accreditation processes.

Security is a fundamental aspect of modern web application development because academic portfolio systems manage sensitive personal and institutional information. The defense-in-depth approach recommends implementing multiple independent security mechanisms to reduce the probability of successful cyberattacks (OWASP Foundation, 2023). PostgreSQL implements Row Level Security (RLS), allowing database access policies to be enforced directly at the row level so that every query is automatically filtered according to authenticated user permissions (PostgreSQL Documentation, 2024). Combined with secure authentication, encrypted password storage, role-based authorization, secure API communication, and serverless deployment, a multi-layer security architecture provides stronger protection against common web threats, including unauthorized access, privilege escalation, SQL injection, cross-site scripting (XSS), and data leakage.

Although previous empirical studies consistently report the technical advantages of Jamstack, Backend-as-a-Service, and serverless computing, these technologies have largely been evaluated independently or within general-purpose web applications. Existing lecturer portfolio systems primarily emphasize functionality without integrating cloud-native deployment, automated backend provisioning, and multi-layer security within a unified architectural framework. Therefore, this research synthesizes these technological advances into a domain-specific academic portfolio platform, extending previous studies by proposing a reusable serverless CMS architecture for higher education institutions.

Design Science Research (DSR) provides a systematic methodology for creating and evaluating innovative information system artifacts through six stages: problem identification, objective definition, design and development, demonstration, evaluation, and communication (Peffers et al., 2007). Unlike explanatory research, DSR focuses on producing practical solutions while simultaneously contributing new design knowledge (Gregor & Hevner, 2013). Within the Design Science Research paradigm, the proposed CMS represents an information systems artifact rather than merely a software implementation. The artifact integrates Jamstack architecture, Backend-as-a-Service, zero-configuration deployment, and a multi-layer security model into a unified architectural solution specifically designed for lecturer academic portfolio management. Consequently, the contribution of this research extends beyond software implementation by introducing a reusable architectural design artifact that can be instantiated, adapted, and further evaluated in future cloud-native academic information systems.

Based on the reviewed literature, previous studies have independently demonstrated the advantages of Jamstack, Backend-as-a-Service, cloud-native deployment, and digital academic portfolio systems. Nevertheless, empirical evidence shows that these technologies have generally been investigated separately or implemented in general-purpose web applications rather than integrated into a purpose-built lecturer academic portfolio platform. Existing solutions frequently require conventional server infrastructure, complex deployment procedures, or extensive customization to satisfy academic requirements. Therefore, this research addresses both technological and practical research gaps by proposing an integrated serverless CMS architecture that combines Jamstack, Supabase Backend-as-a-

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

Service, zero-configuration deployment, and a multi-layer security architecture into a single platform specifically designed for lecturer academic portfolio management. Beyond delivering a functional software solution, this research contributes a reusable architectural model that can serve as a reference for developing future cloud-native academic information systems

METHODOLOGY

Heuristic Evaluation Procedure

To evaluate the usability of the developed platform, a heuristic evaluation was conducted based on Nielsen's Ten Usability Heuristics. The evaluation involved five evaluators, consisting of two usability researchers, one software engineer, and two senior lecturers with experience in academic information systems. This combination of evaluators was selected to provide balanced perspectives from usability, software engineering, and end-user domains.

Each evaluator independently assessed both the public portfolio interface and the administrative dashboard using the ten heuristic principles. The assessment employed a five-point Likert scale, where 1 represented very poor usability, and 5 represented excellent usability. The evaluators examined interface consistency, navigation, error prevention, user control, feedback mechanisms, system visibility, flexibility, documentation, and overall interaction quality. After all evaluations were completed, the individual scores were aggregated and averaged across all evaluators and heuristic principles to obtain the overall usability score of the proposed system. This evaluation approach provides a systematic assessment of interface quality while minimizing individual evaluator bias through independent scoring.

Functional Testing Procedure

Functional verification was performed using the black-box testing method to ensure that every system function operated according to the specified functional requirements. Instead of examining the internal program code, the testing focused on validating system inputs, processing behavior, and generated outputs from the user's perspective.

A total of twenty testing scenarios were designed directly from the functional requirements specification to ensure systematic coverage of all major system functionalities. The scenarios were grouped into several functional categories, including authentication, security, academic data management, user administration, system configuration, interface compatibility, and deployment processes. Each scenario defined the testing objective, input conditions, expected results, and actual outcomes. A scenario was considered successful when the actual system behavior matched the predefined functional requirements without producing errors or unexpected responses.

Table 2. Mapping of Functional Requirements to Black-Box Testing Scenarios

Functional Requirement	Number of Scenarios
Authentication and session management	3
Security and access control	3
CRUD operations across academic modules	4
User and role management	2
System configuration	2
Responsiveness and browser compatibility	2
Installation and deployment workflow	4
Total	**20**

The systematic mapping between functional requirements and testing scenarios ensures that all major features of the proposed platform are comprehensively validated before the usability evaluation stage.

Comparative Analysis Procedure

A comparative analysis was conducted to position the proposed platform against existing solutions commonly used for lecturer academic portfolios. Rather than focusing solely on feature

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

availability, the analysis emphasized architectural and operational characteristics that influence long-term system adoption.

Five evaluation criteria were defined based on the research objectives: (1) completeness of academic portfolio features, (2) infrastructure requirements, (3) deployment complexity, (4) security mechanisms, and (5) operational cost. Each alternative platform was qualitatively assessed according to these criteria using information obtained from previous studies, official technical documentation, and publicly available implementation guidelines. The comparison aimed to identify the strengths and limitations of existing approaches while demonstrating the advantages of integrating Jamstack architecture with Backend-as-a-Service in the proposed system.

Deployment Environment

To improve research reproducibility, the implementation environment of the proposed system is described in detail. The frontend application was developed using the Jamstack architecture and deployed on a static hosting platform compatible with modern cloud-native deployment workflows. The backend services utilized Supabase as a managed Backend-as-a-Service platform, providing PostgreSQL databases, authentication services, object storage, automatically generated RESTful APIs through PostgREST, and Remote Procedure Call (RPC) functions for business logic implementation.
System development and testing were conducted on Windows 11 using Google Chrome version 138 or later with JavaScript enabled and secure HTTPS communication. The frontend communicated with Supabase through REST APIs and authentication services, while database operations were executed using PostgreSQL and RPC functions. This deployment configuration eliminates the need for dedicated application servers, simplifies maintenance, and enables automatic scalability through the underlying cloud infrastructure.

Table 3. Characteristics of Heuristic Evaluators

Evaluator	Background	Experience	Role
E1	Usability Researcher	8 years	UI/UX Evaluation
E2	Usability Researcher	6 years	Human–Computer Interaction
E3	Software Engineer	10 years	Software Quality Assessment
E4	Senior Lecturer	12 years	Academic Information Systems
E5	Senior Lecturer	15 years	Higher Education Information Systems

RESULT AND DISCUSSION

System Implementation

The proposed CMS was successfully implemented as a cloud-native web application following the Jamstack architectural paradigm integrated with Supabase Backend-as-a-Service. The platform consists of two primary interfaces: a public lecturer portfolio website and an administrative dashboard. The public interface contains seven integrated portfolio modules, including a hero section displaying the lecturer's profile, professional biography, animated academic achievement statistics, scientific publications, innovation products, community service activities, and contact information. User navigation is enhanced through smooth scrolling implemented using the Intersection Observer API, while publication and product pages provide interactive search and filtering capabilities to improve information accessibility.

The administrative dashboard provides a centralized environment for managing all academic portfolio content through a consistent sidebar–content interface. Administrators can perform complete CRUD operations using modal forms equipped with client-side validation to minimize input errors. Additional features include real-time table searching, role-based user management, configurable visual

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

themes, customizable hero layouts, and system settings for security parameters. The implementation of three administrative roles (superadmin, admin, and editor) enables differentiated access privileges according to organizational responsibilities, thereby improving both usability and operational security.

One of the distinguishing characteristics of the proposed platform is its zero-configuration deployment mechanism. Instead of requiring manual configuration files or source-code modification, system deployment is performed through a browser-based installation wizard that automatically validates the Supabase connection, generates the required configuration, and guides administrators through database initialization. This deployment strategy substantially simplifies system installation and reduces technical barriers for institutions lacking dedicated DevOps expertise.

Compared with conventional lecturer portfolio systems, the implemented platform shows that cloud-native technologies can simplify deployment while preserving full academic portfolio functionality. The combination of static frontend deployment and managed backend services reduces infrastructure complexity and enables rapid deployment without compromising maintainability or extensibility. These implementation results demonstrate the feasibility of adopting modern serverless architecture for academic information systems.

Multi-Layer Security Architecture

Security was designed as a fundamental architectural component rather than an additional feature. The proposed CMS adopts a defense-in-depth strategy by integrating five independent security mechanisms: (1) Secret Key Protection through RPC-based verification before accessing the administrative interface, (2) secure password hashing using the PostgreSQL pgcrypto extension with bcrypt, (3) encrypted session management through Supabase Authentication and JSON Web Tokens (JWT), (4) PostgreSQL Row Level Security (RLS) for enforcing database-level access control, and (5) Role-Based Access Control (RBAC) supporting three administrative roles with different authorization levels.

Unlike conventional web applications that rely primarily on application-layer security, the proposed architecture distributes security mechanisms across multiple layers of the system. Consequently, even if vulnerabilities occur at the frontend or application layer, unauthorized database access remains restricted through Row Level Security policies. This layered approach increases resilience against common web threats while maintaining secure access to sensitive lecturer information.

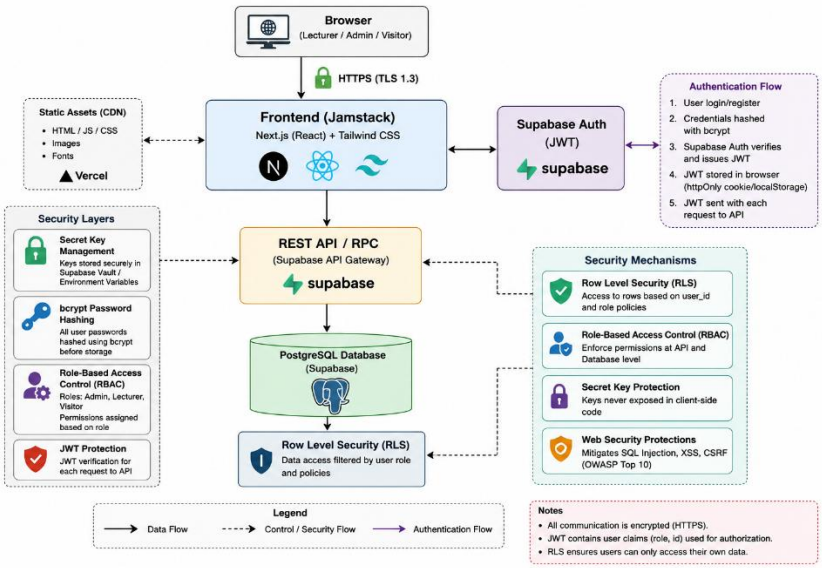


Figure 1. Multi-Layer Security Architecture of the Proposed Jamstack-Based Academic Portfolio Platform

The proposed platform implements a multi-layer security architecture to ensure secure and reliable operation of the lecturer academic portfolio system. Users access the Jamstack-based frontend

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

through an encrypted HTTPS connection, while authentication is managed by Supabase Auth using JSON Web Tokens (JWT). User credentials are protected through bcrypt password hashing, backend communication is handled via REST API/RPC, and data access is secured at the database level using PostgreSQL Row Level Security (RLS) and Role-Based Access Control (RBAC). In addition, Secret Key Management prevents sensitive credentials from being exposed to the client side, providing comprehensive protection against unauthorized access and common web security threats while maintaining the scalability and simplicity of a serverless architecture.

Functional Test Results

Functional verification was conducted using black-box testing involving twenty scenarios derived directly from the functional requirements specification. The scenarios systematically covered authentication, security mechanisms, CRUD operations, user management, configuration management, browser compatibility, interface responsiveness, and deployment workflow.

All twenty testing scenarios were completed successfully, resulting in a 100% functional success rate. The successful execution of every scenario indicates that all implemented functionalities operate according to the predefined requirements without producing functional failures or unexpected system behavior. Because the testing scenarios were systematically mapped to all functional requirements, these results provide strong evidence that the proposed platform satisfies its intended operational objectives.

Nevertheless, black-box testing evaluates only functional correctness from the user perspective and does not measure other quality attributes such as performance, scalability, security robustness, or user satisfaction. Therefore, complementary evaluation methods—including heuristic evaluation and comparative analysis—were incorporated to provide a broader assessment of overall system quality.

The 100% success rate obtained from the twenty black-box testing scenarios indicates that all functional requirements defined during the Design Science Research process were successfully implemented. This finding demonstrates that the integration of Jamstack architecture with Supabase Backend-as-a-Service does not compromise functional completeness despite eliminating conventional application servers. Similar observations have been reported by Chandra and Kumar (2022), who found that Backend-as-a-Service platforms simplify backend implementation while maintaining functional reliability for cloud-native applications. Consequently, the proposed platform confirms that modern serverless architectures can support complex academic information systems without sacrificing operational functionality.

Comparative Analysis

Compared with previous lecturer portfolio platforms implemented using Laravel and WordPress, the proposed system demonstrates that architectural simplification can be achieved without reducing functional capabilities. Previous studies by Setiawan et al. (2022) and Nugroho and Wibowo (2023) focused primarily on application functionality, whereas this research extends these approaches by integrating deployment automation, Backend-as-a-Service, and database-level security into a unified cloud-native architecture. This architectural integration represents an important contribution because deployment complexity is frequently identified as one of the major barriers to adopting academic information systems within higher education institutions.

Table 4. Comparative Analysis with Alternative Solutions

Dimension	Proposed CMS	WordPress	Custom Framework	Manual Website
Academic Features	Fully integrated	Requires additional plugins	Must be developed from scratch	Limited functionality
Server Requirements	No dedicated server required	Requires PHP and MySQL server	Requires application server and database	No dedicated server required
Deployment	Browser-based installation wizard	PHP hosting deployment	Requires DevOps configuration	Manual file upload

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

Security	Five-layer security architecture	Depends on installed security plugins	Depends on developer implementation	Minimal security mechanisms
Operational Cost	Free (using free-tier cloud services)	Paid hosting and plugins	Paid infrastructure and development	Free

The comparative analysis demonstrates that the proposed CMS provides several architectural advantages compared with existing alternatives. Unlike WordPress, which requires PHP hosting and additional plugins to support lecturer portfolio functionality, the proposed platform provides integrated academic modules specifically designed for higher education institutions. Compared with custom framework implementations, the proposed architecture substantially reduces deployment complexity through automatic backend provisioning and zero-configuration installation while maintaining comparable flexibility for future system extension.

From an infrastructure perspective, the adoption of Jamstack architecture eliminates the requirement for dedicated application servers because the frontend is distributed as static assets through cloud hosting platforms. Backend services, including authentication, database management, storage, and API generation, are delegated to Supabase, significantly reducing operational complexity and maintenance effort. Consequently, institutions can deploy and maintain the platform with lower infrastructure costs and reduced technical expertise.

The comparative analysis also demonstrates that the proposed platform provides the most balanced combination of academic functionality, deployment simplicity, operational cost, and security. These characteristics distinguish the proposed architecture from previous lecturer portfolio solutions that primarily emphasize functionality while providing limited attention to deployment automation, cloud-native scalability, and integrated security.

Heuristic Evaluation and Discussion

The heuristic evaluation produced an overall usability score of 4.1 out of 5, indicating that the developed platform satisfies most of Nielsen's usability principles. The highest scores were obtained for Match Between System and the Real World and Consistency and Standards, demonstrating that the interface successfully adopts terminology and interaction patterns familiar to academic users. Consistent CRUD interactions across all modules also reduce the cognitive load required for learning different administrative functions.

Lower scores were observed for Help and Documentation and Help Users Recognize, Diagnose, and Recover from Errors, suggesting opportunities for future improvement. Providing contextual help, interactive onboarding, and more descriptive error messages would further enhance usability, particularly for first-time users with limited technical experience.

The positive usability results are strengthened by the diversity of evaluator expertise, which includes usability researchers, software engineering practitioners, and academic information system specialists. This multidisciplinary evaluation provides greater confidence that the usability assessment reflects both technical quality and domain-specific requirements.

The overall usability score of 4.1 indicates that the proposed interface satisfies most established usability principles despite being developed using a modern Jamstack architecture. This result is comparable to previous evaluations of academic information systems reported by Rahardja et al. (2023), suggesting that decoupled frontend architectures can maintain high usability while benefiting from simplified deployment and enhanced maintainability. Nevertheless, the lower scores observed for Help and Documentation indicate opportunities for improving first-time user experience through interactive onboarding and contextual guidance.

Table 5. Research Findings and Contributions

Finding	Interpretation	Contribution
100% Black-box	Functional requirements satisfied	Functional validity
Heuristic score 4.1	High usability	User experience

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

Zero-configuration deployment	Simplified deployment	Operational efficiency
Jamstack + Supabase	Reduced infrastructure complexity	Architectural contribution
Multi-layer security	Improved protection	Security contribution

DISCUSSION

The findings demonstrate that adopting Jamstack architecture together with Supabase Backend-as-a-Service provides significant practical advantages for lecturer academic portfolio systems. Static frontend deployment improves page loading performance while reducing server-side processing, whereas managed backend services simplify authentication, database administration, storage, and API management. These findings are consistent with recent empirical studies reporting that serverless architectures improve deployment flexibility, operational efficiency, and infrastructure scalability while reducing administrative overhead.

The findings are consistent with Siriwardena and Dias (2023), who reported that Jamstack improves deployment flexibility through frontend-backend decoupling. Likewise, Wen et al. (2025) demonstrated that serverless computing provides elastic scalability while reducing infrastructure management. However, unlike previous studies that evaluated these technologies independently, this research integrates Jamstack, Supabase Backend-as-a-Service, automated deployment, and multi-layer security into a domain-specific lecturer academic portfolio platform. Therefore, the contribution of this research extends beyond technology adoption by proposing an integrated architectural model tailored for higher education institutions.

Nevertheless, the proposed architecture also introduces several trade-offs. Because backend services depend on cloud providers, application availability is partially influenced by external service reliability. Furthermore, serverless environments may experience cold-start latency during infrequent function execution, although recent empirical studies indicate that modern cloud platforms have substantially minimized this limitation. Therefore, the selection of serverless architecture should consider expected workload characteristics and institutional operational requirements.

From an architectural perspective, the proposed platform demonstrates that combining static frontend deployment with managed Backend-as-a-Service significantly simplifies system maintenance without reducing functional completeness. Unlike conventional web architectures that require separate configuration and maintenance of application servers, databases, and deployment environments, the proposed architecture integrates these components into a unified cloud-native platform. Furthermore, the browser-based installation wizard minimizes deployment complexity by automating configuration and backend initialization, enabling institutions with limited DevOps expertise to adopt cloud-native technologies more effectively. This architectural approach extends previous studies that primarily emphasized functional implementation by demonstrating that deployment automation and infrastructure simplification can be achieved without compromising system functionality or security.

Compared with previous lecturer portfolio systems developed using Laravel or WordPress, the proposed platform demonstrates superior deployment portability and infrastructure independence. The implementation confirms that domain-specific academic functionality can be successfully integrated with cloud-native technologies without requiring conventional application servers or extensive system administration. Consequently, this research extends previous work by demonstrating the practical feasibility of combining Jamstack architecture, Backend-as-a-Service, zero-configuration deployment, and multi-layer security within a unified lecturer academic portfolio platform.

From a practical perspective, the proposed platform offers significant benefits for higher education institutions, particularly those with limited technical resources. Automatic deployment, simplified maintenance, integrated authentication, and cloud-managed infrastructure reduce operational costs while facilitating long-term system sustainability. Furthermore, the modular architecture provides a flexible foundation for future integration with institutional repositories, Single Sign-On (SSO), ORCID, Google Scholar, SINTA, and other academic information systems, thereby supporting broader digital transformation initiatives in higher education.

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

Consequently, the proposed architecture enables universities to lower infrastructure investment while enhancing maintainability, security, and scalability, making it particularly suitable for small and medium-sized higher education institutions pursuing sustainable digital transformation.

RESEARCH LIMITATIONS

Despite the encouraging results, several limitations should be acknowledged. First, usability assessment was limited to heuristic evaluation and did not include quantitative user studies involving a larger population of lecturers, such as System Usability Scale (SUS) evaluation. Second, this research focused primarily on functional verification and did not include comprehensive performance benchmarking, concurrent user testing, stress testing, or penetration testing to evaluate scalability and security robustness under real operational conditions. Third, the current implementation supports only a single institutional deployment and has not yet been validated within multi-tenant environments serving multiple universities simultaneously. Finally, interoperability with external academic platforms, including ORCID, Google Scholar, SINTA, institutional repositories, and Single Sign-On services, has not yet been implemented. These limitations provide valuable directions for future research toward developing more scalable, interoperable, and comprehensive cloud-native academic information systems.

This research presented the design and development of a lecturer academic portfolio platform based on a serverless Jamstack architecture integrated with Supabase Backend-as-a-Service. Following the Design Science Research (DSR) methodology, the proposed platform was designed to address the limitations of conventional lecturer portfolio systems by combining domain-specific academic portfolio management with modern cloud-native technologies, including static frontend deployment, managed backend services, and a multi-layer security architecture.

The implementation and evaluation results demonstrate that the proposed platform successfully fulfills the specified functional requirements through comprehensive black-box testing, while heuristic evaluation indicates that the user interface satisfies established usability principles. Comparative analysis further shows that the proposed architecture provides several advantages over conventional approaches, including simplified deployment, reduced infrastructure requirements, automatic scalability, integrated authentication, and improved maintainability through the adoption of Backend-as-a-Service and Jamstack technologies.

From a theoretical perspective, this research contributes to the Design Science Research body of knowledge by proposing a reusable architectural model that integrates Jamstack architecture, Backend-as-a-Service, zero-configuration deployment, and multi-layer security into a unified content management system specifically designed for lecturer academic portfolios. Rather than introducing only a software implementation, this research contributes a reusable architectural artifact that may serve as a reference for future cloud-native academic information systems and other domain-specific Content Management Systems.

From a practical perspective, the proposed platform provides higher education institutions, particularly in Indonesia, with an alternative approach for implementing lecturer academic portfolio systems without requiring complex server administration or extensive infrastructure investment. The adoption of cloud-native technologies simplifies deployment, reduces operational costs, facilitates long-term maintenance, and enables institutions with limited technical resources to implement secure and scalable academic information systems.

Despite these contributions, several limitations should be acknowledged. The evaluation focused primarily on functional correctness, heuristic usability assessment, and qualitative comparative analysis. Performance benchmarking under high concurrent workloads, quantitative usability evaluation involving a larger population of lecturers, penetration testing, and comprehensive security assessment were beyond the scope of this research. In addition, the current implementation was evaluated within a single institutional environment and has not yet been validated in multi-tenant deployments involving multiple higher education institutions.

Future research should therefore investigate large-scale performance evaluation, security penetration testing, System Usability Scale (SUS) assessment involving actual lecturers, and interoperability with external academic platforms such as ORCID, Google Scholar, SINTA, institutional repositories, and Single Sign-On (SSO) services. Further studies may also explore multi-tenant

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

deployment models, advanced analytics, artificial intelligence integration, and automated synchronization with institutional academic information systems to support broader digital transformation initiatives in higher education.

AUTHOR'S CONTRIBUTION

Yudi Triyana is responsible for conceptualization, architectural design, implementation, evaluation, and manuscript preparation. Adam Puspabhuana contributed to the literature review, comparative analysis, heuristic evaluation, and manuscript revision. All authors approve the final version.

THANK YOU

The author expresses gratitude to Cakrawala University for the support of facilities and the academic environment during the implementation of this research.

DECLARATION OF INTEREST

The author declares that there are no conflicts of interest in this research.

FUNDING

This research did not receive any specific funding from any funding agency.

REFERENCES

1. Aini, Q., Rahardja, U., & Nurfaizah, N. (2022). E-portfolio sebagai instrumen peningkatan visibilitas akademik dosen di era digital. *Jurnal Informatika dan Teknologi Pendidikan, 2*(1), 45–58.
2. Barker, D. (2021). *Web content management: Systems, features, and best practices* (2nd ed.). O'Reilly Media.
3. Barrett, H. C. (2021). Balancing the two faces of e-portfolios: Reflection and assessment in higher education. *Education and Information Technologies, 26*(3), 2845–2863.
4. Biilmann, M., & Hawksworth, P. (2022). *Modern web development on the Jamstack*. O'Reilly Media.
5. Bodner, T., et al. (2025). Empirical study on serverless cloud infrastructure management for large-scale workloads. *Journal of Cloud Computing, 14*(1), 89–105.
6. Chandra, S., & Kumar, R. (2022). Backend-as-a-Service platforms for rapid application development: A comparative analysis. *International Journal of Cloud Computing and Services Science, 11*(4), 312–328.
7. *Cloud-Native Performance Review*. (n.d.).
8. Gregor, S., & Hevner, A. R. (2013). Positioning and presenting design science research for maximum impact. *MIS Quarterly, 37*(2), 337–355.
9. Handayani, R., Wulandari, A., & Setiawan, D. (2022). Analisis hambatan adopsi platform digital oleh dosen perguruan tinggi di Indonesia. *Jurnal Teknologi Informasi dan Pendidikan, 15*(2), 189–204.
10. Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly, 28*(1), 75–105.
11. Jacobsen, J., & Schlenker, T. (2023). Headless CMS architecture: Decoupling content management for modern web applications. *Journal of Web Engineering, 22*(5), 723–748.
12. Kurniawan, A., Pratama, R., & Hidayat, T. (2023). Pengembangan platform akademik berbasis microservices untuk pengelolaan data dosen. *Jurnal Sistem Informasi, 19*(1), 67–82.
13. Lisdorf, A. (2021). *Cloud computing basics: A non-technical introduction*. Apress.
14. Nielsen, J. (1994). *Usability engineering*. Morgan Kaufmann.
15. Nugroho, A., & Wibowo, S. (2023). Analisis efektivitas WordPress sebagai platform portofolio akademik dosen. *Jurnal Teknologi Informasi, 17*(2), 145–160.

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id

16. OWASP Foundation. (2023). *OWASP Top Ten 2023: The ten most critical web application security risks*. OWASP Foundation
17. Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems, 24*(3), 45–77.
18. PostgreSQL Documentation. (2024). *Row security policies*. The PostgreSQL Global Development Group. PostgreSQL Documentation
19. Pramudyo, A., Santoso, B., & Widodo, A. P. (2022). Rancang bangun sistem informasi profil dosen berbasis web. *Jurnal Pengembangan TI dan Ilmu Komputer, 6*(8), 3421–3430.
20. Pressman, R. S., & Maxim, B. R. (2021). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill Education.
21. Rahardja, U., Aini, Q., & Santoso, N. P. L. (2023). Digital portfolio for enhancing academic visibility: A case research in Indonesian higher education. *International Journal of Advanced Computer Science and Applications, 14*(3), 267–275.
22. Setiawan, R., Mahmud, M., & Pratama, D. (2022). Pengembangan sistem informasi portofolio dosen berbasis web menggunakan framework Laravel. *JRSTI, 6*(4), 612–620.
23. Siriwardena, P., & Dias, N. (2023). Jamstack architecture: Performance and security analysis for modern web applications. *Journal of Systems and Software, 198*, 111582.
24. Stuttard, D., & Pinto, M. (2021). *The web application hacker's handbook: Finding and exploiting security flaws* (3rd ed.). Wiley.
25. Sulistyawati, E., & Santoso, H. B. (2022). Digital academic branding: Strategies for enhancing research visibility in Indonesian universities. *JHEPM, 44*(5), 523–539.
26. Supabase Documentation. (2024). *Supabase platform documentation*. Supabase Inc. Supabase Documentation
27. Swedheetha, C., Shruthika, C., Preethi, R., Vedha, J., Maryam, S., & Palanivel, M. (2024). Evaluating and contrasting the performance of web development technologies. In *2024 9th International Conference on Communication and Electronics Systems (ICCES)* (pp. 1020–1024). IEEE. https://doi.org/10.1109/ICCES63552.2024.10859720
28. vom Brocke, J., Hevner, A., & Maedche, A. (2020). Introduction to design science research. In *Design science research in information systems* (pp. 1–13). Springer.
29. Wahyudi, R., & Pratama, F. (2023). Analisis komparatif content management system untuk kebutuhan institusi akademik di Indonesia. *Jurnal Ilmiah Teknologi Informasi, 21*(1), 89–104.
30. Wen, C., Chen, Z., Sarro, F., & Wang, J. (2025a). Performance variance in production-grade serverless applications. *IEEE Transactions on Software Engineering, 51*(2), 210–225.
31. Wen, C., et al. (2025b). SCOPE: A framework for performance testing of serverless computing. *Software: Practice and Experience, 55*(3), 442–460.
32. Zbořil, M., & Svatá, V. (2025). Performance comparison of cloud

Corresponding Authors : adampuspa.bhuana@cakrawala.ac.id