A Lightweight Web Push Notification Framework for Educational Information Systems

Supporting Digital Transformation Through Real-Time Web Notification Technology

Luqman Hakim[1)]

[1)]*Program Studi Teknik Informatika, Fakultas Teknik, Universitas Bakti Indonesia*
Jalan Kampus Bumi Cempokosari No.40, Cluring - Banyuwangi 68482

luqman@ubibanyuwangi.ac.id

Abstract

This study presents the design and implementation of a lightweight Web Push Notification Framework for educational information systems using PHP Native, JavaScript, Service Worker, Web Push API [1]*, VAPID authentication, and MySQL. The proposed framework consists of four core modules: Service Worker* [2]*, Push Subscription, Subscription Management, and Notification Delivery, enabling secure browser registration and automatic real-time notification delivery whenever new content is published. Experimental results confirm that the framework successfully performs browser registration, subscription storage, notification delivery, and content redirection across modern web browsers. The proposed framework provides a secure, scalable, and easily integrated solution for PHP-based web applications while establishing a foundation for future intelligent notification services.*

Keywords*: Web Push Notification, PHP Native, Service Worker, Web Push API, VAPID Authentication, Educational Information System.*

Introduction

a. **General Background**. Educational institutions increasingly rely on web-based information systems to disseminate academic announcements, news, and institutional activities. School web portals have become an essential communication medium for students, parents, and staff due to their accessibility and ease of deployment. However, most web portals remain passive communication platforms, requiring users to manually access the website to obtain the latest information. This approach often delays information dissemination and reduces user engagement.

Recent advancements in web technologies, particularly the Web Push API [1], Notification API, and Service Worker, have enabled websites to deliver real-time notifications directly to users' browsers without requiring dedicated mobile applications. These standardized technologies provide an efficient solution for improving information accessibility while maintaining compatibility with modern web browsers.

b. **Problem Statement**. Although many educational institutions have adopted web-based information systems, real-time information delivery remains a challenge. Users must frequently revisit the website to check for new announcements, while administrators often depend on external messaging platforms that are not fully integrated with institutional systems. Consequently, important information may be delivered late or overlooked.

Furthermore, implementing Web Push Notification using native web technologies is often considered complex due to the integration of browser APIs, Service Workers [2], VAPID authentication [3], subscription management, and backend

Submitted: 16 Jan 2026; Revised: 16 Jan 2026; Accepted: 16 Jan 2026 Published: 16 Jan 2026

notification delivery. As a result, many PHP-based web applications still lack a practical and reusable notification framework.

c. **Existing Solution**. Existing push notification solutions commonly utilize cloud-based services such as Firebase Cloud Messaging (FCM) and One Signal, which simplify deployment through managed infrastructure. Although these platforms reduce development effort, they introduce dependencies on third-party services and provide limited flexibility for customization and long-term maintenance.
Alternatively, the standardized Web Push API with VAPID authentication [3] enables secure browser-based notifications without proprietary messaging SDKs. However, current implementations are generally presented as application-specific solutions rather than reusable frameworks that can be easily integrated into PHP-based information systems.

d. **Research Gap**. Previous studies have primarily focused on implementing Web Push Notification as an application feature rather than providing a reusable implementation framework. Most existing solutions emphasize notification delivery while offering limited guidance on integrating browser subscription management, backend processing, database storage, and secure notification delivery into PHP-based web applications.
In addition, existing implementations rarely address framework modularity, scalability, and maintainability, making it difficult to adopt across different educational information systems. These limitations highlight the need for a lightweight and standardized implementation framework based entirely on native web technologies.

e. **Proposed Solution**. This research proposes a lightweight Web Push Notification Framework [1] for educational information systems using PHP Native, JavaScript [4], Service Worker [2], Web Push API, MySQL, and VAPID authentication [3]. The framework is organized into four modular components: Service Worker [2], Push Subscription, Subscription Management, and Notification Delivery.
By adopting a modular architecture [5], the proposed framework simplifies the integration of secure real-time notifications into existing PHP-based web applications while remaining independent of proprietary cloud messaging SDKs.

f. **Research Contribution**. The main contributions of this study are summarized as follows:
- Proposing a lightweight and modular Web Push Notification Framework for PHP-based educational information systems.
- Designing four reusable framework components covering browser subscription, subscription management, notification delivery, and Service Worker [2] implementation.
- Implementing secure browser-to-server communication using the Web Push API and VAPID authentication [3].
- Providing a practical implementation reference that simplifies the integration of real-time notifications into existing PHP applications.
- Establishing a scalable foundation for future extensions, including personalized notifications, analytics, and intelligent notification management.

Related Works

Recent studies have demonstrated the effectiveness of Web Push Notification technology in improving real-time information dissemination and user engagement across web applications [6]. Most implementations rely on cloud-based services such as Firebase Cloud Messaging (FCM) and OneSignal, which simplify deployment through managed infrastructures and developer-friendly APIs. However, these approaches introduce dependencies on third-party services and provide limited flexibility for infrastructure management and long-term customization.

Submitted: 16 Jan 2026; Revised: 16 Jan 2026; Accepted: 16 Jan 2026 Published: 16 Jan 2026

With the standardization of modern web technologies, browsers now natively support the Web Push API, Notification API, and Service Worker, enabling secure browser-based notifications without dedicated mobile applications. Combined with VAPID authentication, these technologies provide a standardized mechanism for secure communication between application servers and web browsers while reducing reliance on proprietary notification platforms.

Although Web Push Notification has been implemented in various application domains [7], most existing implementations are designed for specific applications rather than as reusable implementation frameworks. As a result, integrating browser subscription management, backend processing, and notification delivery into PHP-based applications still requires significant development effort.

To address this limitation, this study proposes a lightweight Web Push Notification Framework [5] that organizes the implementation into four reusable modules: Service Worker, Push Subscription, Subscription Management, and Notification Delivery. The proposed framework provides a modular implementation that can be easily integrated into PHP-based educational information systems using standardized web technologies.

Table I summarizes the characteristics of existing implementation approaches and highlights the contribution of the proposed framework.

Table I. Literature comparison table

Aspect	**Existing Approaches**	**Proposed Framework**
Implementation	Application-specific	Modular framework
Platform	Mostly cloud-based	Native PHP
Notification	Web Push	Web Push
Authentication	Platform-dependent	VAPID
Integration	Limited	Easy integration into PHP systems
Reusability	Low	High

Proposed Framework Architecture

The proposed Web Push Notification framework provides secure and real-time information delivery through a lightweight and modular architecture. The framework consists of three main layers: the client side, the application server, and the database, and is organized into four functional modules: Service Worker, Push Subscription, Subscription Management, and Notification Delivery.

On the client side, users access the educational web portal through a modern web browser. After granting notification permission, the browser registers a Service Worker and creates a push subscription using the Web Push API. The generated subscription data are securely transmitted to the application server and stored in the database for future notification delivery.

The application server, implemented using PHP Native, manages news content, browser subscriptions, and notification requests. Whenever a new article or announcement is published, the server retrieves all active subscriptions from the database and securely delivers notification payloads using VAPID authentication [3] over the Web Push Protocol. This mechanism establishes secure communication between the application server and the browser push service.

Upon receiving a push event, the registered Service Worker processes the notification and displays it through the Notification API. Selecting the notification redirects users to the corresponding news article or announcement page.

By adopting standardized web technologies, the proposed framework provides a lightweight, modular, and reusable architecture that can be easily integrated into PHP-based educational information systems without relying on proprietary cloud messaging platforms.

System Design and Implementation

a. **System Design**. The proposed Web Push Notification framework adopts a lightweight client–server architecture organized into four functional modules: Service Worker [2], Push Subscription, Subscription Management, and

Submitted: 16 Jan 2026; Revised: 16 Jan 2026; Accepted: 16 Jan 2026 Published: 16 Jan 2026

Notification Delivery [8], as illustrated in Figure 1.

Users first grant notification permission through a web browser, where the Service Worker is registered and a push subscription is created using the Web Push API. The subscription information is then transmitted to the Subscription Management module, which validates and stores the subscription data in the MySQL database.

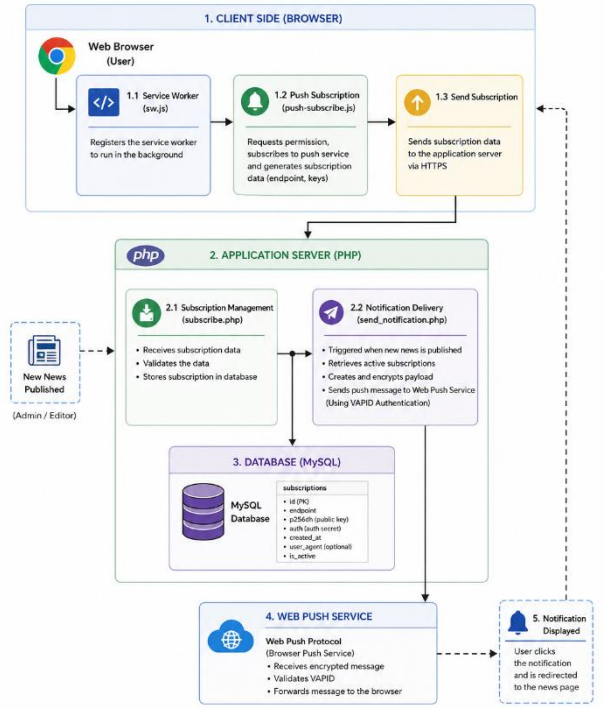

Figure 1. Overall System Architecture

Whenever a new article or announcement is published, the Notification Delivery module retrieves all active subscriptions and securely sends notification payloads using the Web Push Protocol with VAPID authentication. The browser receives the push event through the registered Service Worker, which displays the notification and redirects users to the corresponding news page when selected.

This modular architecture provides a lightweight and reusable framework that can be easily integrated into existing PHP-based educational systems while maintaining secure and compliant standard notification delivery.

b. **Browser Subscription**. As illustrated in Figure 2, the subscription workflow begins when users grant notification permission through a supported web browser. The browser then registers a Service Worker and creates a push subscription using the Web Push API. The generated subscription information, including the endpoint and cryptographic keys, is securely transmitted to the application server for further processing.

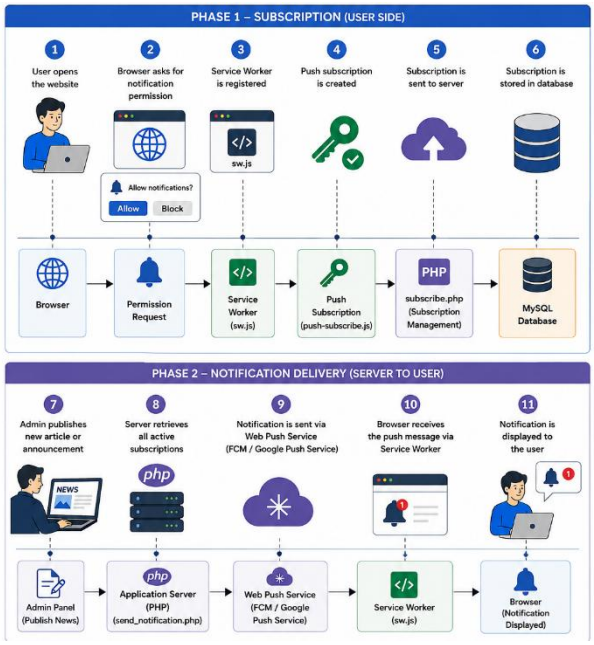

Figure 2 Notification Workflow of the Proposed System [8]

c. **Subscription Management.** Figure 3 illustrates the subscription registration process implemented in the proposed framework. The process begins when a user accesses the educational web portal through a supported web browser. After the user grants notification permission, the browser registers a Service Worker and creates a push subscription using the Web Push API.

The generated subscription object, which contains the endpoint and authentication credentials, is securely transmitted to the Subscription Management module on the application server. The server validates the received subscription data before storing it in the MySQL database. These subscription records are subsequently used

Submitted: 16 Jan 2026; Revised: 16 Jan 2026; Accepted: 16 Jan 2026 Published: 16 Jan 2026

by the Notification Delivery module to identify active subscribers and deliver notifications whenever new content is published.

d. **Notification Delivery**. When an administrator publishes a news article or announcement, the Notification Delivery module retrieves all active subscriptions from the database. Notification payloads are securely transmitted through the Web Push Protocol using VAPID authentication. The registered Service Worker receives the push event and displays the notification to the user, who can then access the corresponding content directly from the browser.

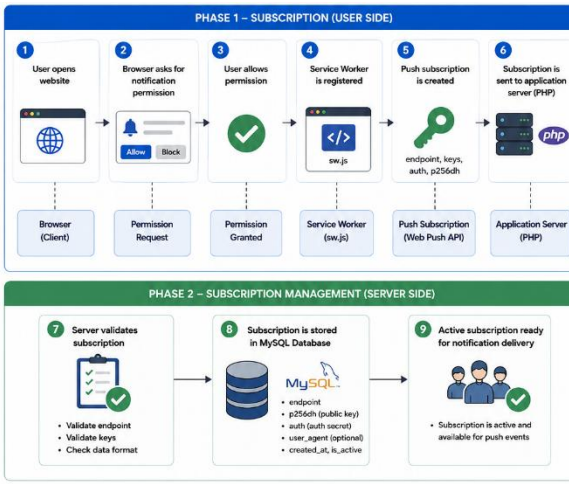

Figure 3. Subscription Registration Process

e. **Notification Display.** Figure 4 demonstrates the notification display process after the browser receives a push event. Once the notification payload is delivered through the Web Push Protocol, the registered Service Worker processes the incoming message and invokes the Notification API to display a browser notification. The notification presents essential information, including the news title, a brief description, and an application icon, allowing users to immediately recognize newly published content.

When the user selects the notification, the browser automatically opens the corresponding news page specified in the notification payload. This mechanism enables users to access newly published articles directly without manually revisiting the website, thereby improving the timeliness and accessibility of information.

```
self.registration.showNotification(title, options);
```

Figure 4. Show Notification

The showNotification() method is invoked by the Service Worker after receiving a push event to display the notification through the browser's Notification API.

f. **Framework Implementation.** The proposed framework was implemented using PHP Native as the application backend and MySQL for subscription storage. On the client side, JavaScript [4], Service Worker, and the Web Push API handle browser registration, push subscription, and notification processing. Secure communication between the application server and browser push service is established through HTTPS and VAPID authentication, ensuring standards-compliant and secure notification delivery. Table II summarizes the implementation environment..

Table II. Framework Implementation Environment

Component	Technology
Backend	PHP 8.x (Native)
Frontend	HTML5, JavaScript (ES6) [4]
Database	MySQL
Browser API	Web Push API
Notification API	Notification API
Background Process	Service Worker
Authentication	VAPID
Transport Security	HTTPS

Experimental Results and Evaluation

Table III summarizes the functional evaluation results of the proposed framework. The evaluation covered all major framework components, including browser permission, Service Worker registration, push subscription creation, subscription storage, notification delivery, browser notification display, and notification redirection. All test scenarios were completed successfully, indicating that each

Submitted: 16 Jan 2026; Revised: 16 Jan 2026; Accepted: 16 Jan 2026 Published: 16 Jan 2026

module operated according to the proposed design.

The evaluation confirms that the proposed framework successfully performs the complete notification workflow, from browser registration and subscription management to secure notification delivery and browser notification display. These results demonstrate that native web technologies can provide secure and real-time information delivery without requiring proprietary cloud messaging platforms.

Overall, the proposed framework provides a lightweight and reusable solution for integrating Web Push Notification into PHP-based educational information systems. Its modular architecture also provides a practical foundation for future enhancements, including notification analytics, user segmentation, and intelligent notification services.

Table III. Framework Evaluation Results

Framework Component	Evaluation Result	Status
Browser Permission	Successful	✓
Service Worker	Registered Successfully	✓
Push Subscription	Created Successfully	✓
Subscription Management	Stored Successfully	✓
Notification Delivery	Delivered Successfully	✓
Browser Notification	Displayed Successfully	✓
Notification Redirection	Successful	✓

Discussion

The experimental results demonstrate that the proposed framework successfully delivers real-time notifications using standardized web technologies. By leveraging the Web Push API, Service Worker, and VAPID authentication, the framework enables secure and automatic notification delivery without requiring dedicated mobile applications or proprietary cloud messaging platforms. This standards-based approach ensures interoperability with modern web browsers while providing a lightweight implementation for educational information systems.

Compared with conventional educational web portals, the proposed framework improves information dissemination by automatically notifying users whenever new content is published. The modular implementation based on PHP Native and MySQL also facilitates integration into existing PHP-based web applications with minimal modifications, making the framework practical for educational environments.

Although the framework successfully implements secure real-time notification delivery, it currently adopts a broadcast notification model in which all subscribed users receive identical notifications. In addition, the framework does not yet support user segmentation, notification analytics, or personalized notification delivery. These limitations provide opportunities for extending the framework in future studies.

Future Works

Several enhancements can be incorporated into the proposed framework to improve its functionality and applicability. First, personalized notification delivery can be introduced by recommending content based on user interests or browsing behavior. Second, notification scheduling may be implemented to optimize delivery time according to user activity patterns.

Future work may also include the development of a notification analytics dashboard to monitor subscription growth, delivery success, click-through rates, and user engagement. In addition, Natural Language Processing (NLP) or Machine Learning techniques may be integrated to support automatic news categorization [9], content prioritization, and personalized notification delivery. These enhancements would extend the proposed framework into a more intelligent and adaptive communication platform for educational information systems.

References

[1] M. Thomson, E. Damaggio, and B. Raymor, “Generic Event Delivery Using HTTP Push,” no. 8030. in Request for

Submitted: 16 Jan 2026; Revised: 16 Jan 2026; Accepted: 16 Jan 2026 Published: 16 Jan 2026

Comments. RFC Editor, Dec. 2016. doi: 10.17487/RFC8030.
[2] T. Sutter, K. Lapagna, P. Berlich, M. Rennhard, and F. Germann, “Web Content Signing with Service Workers.” 2021. [Online]. Available: https://arxiv.org/abs/2105.05551
[3] M. Thomson and P. Beverloo, “Voluntary Application Server Identification (VAPID) for Web Push,” no. 8292. in Request for Comments. RFC Editor, Nov. 2017. doi: 10.17487/RFC8292.
[4] E. Wittern, A. T. T. Ying, Y. Zheng, J. Dolby, and J. A. Laredo, “Statically Checking Web API Requests in JavaScript.” 2017. [Online]. Available: https://arxiv.org/abs/1702.03906
[5] “The Web Push Protocol | Articles,” web.dev. Accessed: Jul. 01, 2026. [Online]. Available: https://web.dev/articles/push-notifications-web-push-protocol
[6] A. Rahmatulloh, A. Rachman, and F. Anwar, “Implementasi Web Push Notification pada Sistem Informasi Manajemen Arsip Menggunakan PUSHJS,” *J. Teknol. Inf. Dan Ilmu Komput.*, vol. 6, p. 327, May 2019, doi: 10.25126/jtiik.201963936.
[7] “Implementasi Web Push Notification pada Sistem Informasi Manajemen Arsip Menggunakan PUSHJS,” *J. Teknol. Inf. Dan Ilmu Komput.*, vol. 6, no. 3, pp. 327–334, May 2019, doi: 10.25126/jtiik.201963936.
[8] “Implementing Push Notifications with the Web Push API.” Accessed: Jul. 01, 2026. [Online]. Available: https://blog.openreplay.com/implementing-push-notifications-web-push-api/
[9] K. Subramani, J. Jueckstock, A. Kapravelos, and R. Perdisci, “SoK: Workerounds - Categorizing Service Worker Attacks and Mitigations,” in *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, IEEE, Jun. 2022, pp. 555–571. doi: 10.1109/eurosp53844.2022.00041.

Submitted: 16 Jan 2026; Revised: 16 Jan 2026; Accepted: 16 Jan 2026 Published: 16 Jan 2026