***American Worldviews: Toward a Sustainable Future (Workshop – March 18–21, 2025, Univer- sidad Tecnológica de El Salvador, San Salvador, El Salvador)}, Vol. 84, 2025***

**University of New Mexico**

Registro de Asistencia mediante una Aplicación Web (PWA)

# Alejandro León-Salazar 1, Natasha Mehler-Castro 1, Arianna Esteves-Valero 1, Nohelya Monar-Zambrano 1, Denisse Crespo-Arias 1

1 Facultad de Ciencias Matemáticas y Físicas, Universidad de Guayaquil, Guayaquil, Ecuador; [xavier.leonsal@ug.edu.ec](mailto:xavier.leonsal@ug.edu.ec), [natasha.mehlercas@ug.edu.ec](mailto:natasha.mehlercas@ug.edu.ec), [arianna.estevesval@ug.edu.ec](mailto:arianna.estevesval@ug.edu.ec), [nohelya.monarzam@ug.edu.ec,](mailto:nohelya.monarzam@ug.edu.ec,) [denisse.crespoari@ug.edu.ec](mailto:denisse.crespoari@ug.edu.ec)

**Resumen:** Este artículo presenta el desarrollo de una aplicación web ligera destinada al registro y consulta de asistencias. El sistema fue concebido para ofrecer una alternativa sencilla y eficiente frente a soluciones comerciales complejas, priorizando la facilidad de uso y la accesibilidad desde cualquier dispositivo con navegador web. La aplicación permite a los usuarios registrar materias, asegurando que los horarios no se solapen, y facilita el control de asistencia diaria. Al acceder, el usuario puede visualizar si tiene clase en ese momento, registrar su asistencia y consultar una tabla semanal que resume los días y materias asistidas, además de obtener un resumen general de su participación. El backend, desarrollado en PHP con base de datos MySQL, garantiza la integridad y seguridad de los datos.

**Palabras Clave:** registro de asistencia, aplicación web, PHP, MySQL, educación digital, instalación web

# Introducción

La gestión eficiente de la asistencia es un aspecto esencial en la administración académica moderna, pues tiene un impacto directo en la organización, el desempeño y la retención estudiantil [1]. En la Universidad de Guayaquil, el programa de Ingeniería de Software enfatiza no solo el dominio técnico, sino también la capacidad de diseñar y construir soluciones que respondan a necesidades reales del entorno educativo. El desarrollo de sistemas prácticos, como aplicaciones para el registro y control de asistencias, permite a los estudiantes aplicar conocimientos sobre diseño de bases de datos, experiencia de usuario, lógica de negocio y buenas prácticas de seguridad [2,3].

El sistema presentado en este trabajo responde a la necesidad de una herramienta sencilla y estandarizada, accesible a cualquier usuario con conocimientos básicos, sin sacrificar seguridad ni eficiencia. Desarrollada con PHP y MySQL como base tecnológica, la aplicación permite gestionar el ciclo completo de registro y consulta de asistencias desde cualquier navegador moderno. Su diseño, ligero y modular, busca reducir la curva de aprendizaje y facilitar su implementación en distintos entornos.

A diferencia de plataformas comerciales ampliamente conocidas como Jibble, Clockify o incluso módulos de asistencia incluidos en suites educativas más amplias, la propuesta se distingue por su enfoque minimalista: elimina funciones innecesarias, prioriza la simplicidad del flujo de uso y asegura que cualquier usuario pueda instalar y comenzar a utilizar la herramienta en minutos, sin dependencias externas ni configuraciones avanzadas. Mientras que otras soluciones suelen estar acotadas a plataformas cerradas, exigir pagos por licencia o requerir instalaciones complejas y mantenimiento especializado [4,5], el sistema desarrollado aquí se limita a lo esencial: el usuario registra sus materias, el sistema valida que los horarios no se solapen, y desde el inicio se le informa si tiene clase en ese momento o si está libre. Posteriormente, puede consultar una tabla semanal que muestra, de lunes a domingo, qué días asistió y en qué materia, así como visualizar un resumen general de sus asistencias. Esta simplicidad no solo facilita el uso y la adopción, sino que también garantiza que el sistema sea eficiente y accesible para cualquier persona que requiera un control básico de su asistencia académica.

En este artículo se presenta el desarrollo y evaluación de un sistema de registro de asistencia académico, llevado a cabo por estudiantes de la Universidad de Guayaquil. El sistema fue concebido como una aplicación web instalable, adaptada a las necesidades de los usuarios finales y enfocada en la simplicidad, la eficiencia y la seguridad. A lo largo del trabajo, se describen los componentes del sistema, su arquitectura, las políticas de tolerancia a fallos, el manual de usuario y los resultados de su validación en un entorno real.

# Materiales y Métodos

* 1. ***Descripción del Sistema***

El sistema consiste en una aplicación web, accesible desde cualquier dispositivo moderno (computadora, tablet o teléfono). Se apoya íntegramente en tecnologías estándar: HTML5, CSS3, JavaScript para la interfaz; PHP para la lógica de servidor; y MySQL como gestor de base de datos relacional.

El sistema permite registrar materias, definir horarios, marcar asistencias diarias y obtener reportes históricos. La arquitectura modular facilita la escalabilidad y el mantenimiento futuro del software.

Figura 1. Diagrama de componentes que describe la arquitectura del sistema.

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

**Figura 1.** Diagrama de componentes del sistema.

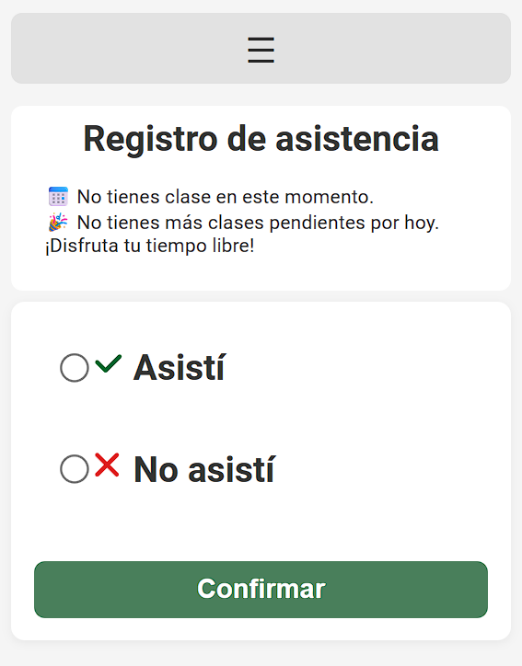
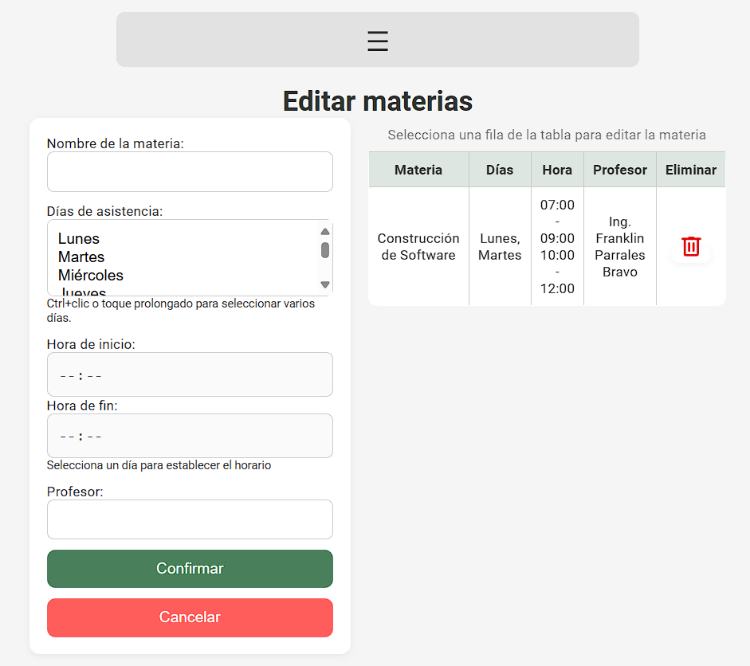
A continuación, se detallan los componentes principales del sistema, agrupados en función de su propósito:

1. Frontend:

Proporciona formularios para la creación de cuentas, registro y edición de materias, y registro de asistencias. Incluye validaciones en tiempo real para mejorar la experiencia y prevenir errores comunes.

1. Backend: Gestiona la lógica de negocio y la seguridad, incluyendo la autenticación de usuarios, la validación de sesiones y la protección de datos sensibles.
2. MySQL: Se emplea como sistema gestor de base de datos para almacenar toda la información relevante de usuarios, materias, horarios y asistencias. MySQL facilita consultas eficientes y garantiza la integridad en un entorno multiusuario.
3. Módulos Adicionales: Se han implementado mecanismos de tolerancia a fallos, manejo de errores, y políticas de mantenimiento preventivo y correctivo, documentadas en los planes respectivos.

En la Figura 2 y 3 se puede ver un ejemplo de la interfaz del sistema Para más detalles sobre su uso, puede consultar el manual de usuario disponible en su repositorio de GitHub [1].



**Figura 2.** Interfaz para registrar la asistencia. **Figura 3.** Interfaz para registrar materias.

**2.2 Arquitectura del Sistema**

El sistema adopta un modelo cliente-servidor clásico. El lado cliente consiste en una interfaz web responsiva construida con HTML5, CSS3 y JavaScript, optimizada para dispositivos móviles y de escritorio. El servidor utiliza PHP para la lógica de negocio y la gestión de sesiones, mientras que MySQL almacena los datos de usuarios, materias, horarios y asistencias. Las comunicaciones entre cliente y servidor emplean AJAX, lo que permite actualizaciones dinámicas sin recargar la página y mejora la experiencia de usuario. Todas las transacciones sensibles requieren autenticación activa, reforzando la privacidad de los datos.

El sistema está diseñado con una arquitectura modular y escalable, permitiendo añadir funcionalidades o modificar componentes con relativa facilidad. La separación clara entre la capa de presentación (frontend) y la lógica de negocio y datos (backend) facilita el mantenimiento y mejora la seguridad.

**2.3 Documentación Técnica**

Durante el desarrollo se elaboraron documentos de requisitos funcionales y no funcionales, alcance, tolerancia a fallos, políticas y plan de mantenimiento, plan de pruebas, diseño detallado de software y manual de usuario. Estas guías permitieron un desarrollo sistemático, minimizando riesgos y mejorando la calidad del producto final.

La tolerancia a fallos incluye validaciones en frontend y backend, manejo de sesiones expiradas y gestión de errores de conexión o de base de datos, garantizando la robustez del sistema en ambientes reales, como recomienda [Sommerville, 2011](https://www.pearson.com/en-us/subject-catalog/p/software-engineering/P200000007433/9780137035151) [6].

**3. Requerimientos del Sistema**

**3.1 Requerimientos Funcionales**

Para el desarrollo de la aplicación, se definieron requerimientos funcionales claros que guían el diseño y la implementación, los cuales aseguran que la aplicación cumpla con las necesidades básicas de los usuarios y el negocio:

* **Gestión de Materias**: Los usuarios deben poder crear, editar y eliminar materias. Cada materia contiene información clave como el nombre, los días en los que se imparten clases, el horario definido y el nombre del profesor responsable. Se implementaron validaciones tanto en el frontend como en el backend.
* **Registro de Asistencia**: Se habilita a los usuarios para registrar su asistencia diaria a las materias correspondientes. El sistema restringe el registro solo dentro de los horarios establecidos para cada materia y evita la duplicidad, de modo que no se puedan registrar varias asistencias para la misma materia en un mismo día.
* **Consulta de Asistencias**: La aplicación provee un resumen visual de las asistencias registradas, con la posibilidad de filtrar la información por mes y por semana, facilitando el seguimiento histórico del cumplimiento de las asistencias.
* **Registro y Autenticación de Usuarios**: Se permite a los usuarios crear una cuenta con sus datos personales básicos: nombre, correo electrónico y contraseña. Se verifica que el correo no esté previamente registrado para evitar duplicados. Las contraseñas son almacenadas de forma segura utilizando funciones hash robustas (como password\_hash() en PHP).
* **Persistencia de Datos y Protección**: Todos los registros de materias, asistencias y usuarios se almacenan en una base de datos MySQL gestionada por PHP, garantizando integridad y seguridad. Se protege el acceso a funciones sensibles a través de validación de sesiones activas, para asegurar que solo usuarios autenticados puedan acceder a ellas.

**3.2 Requerimientos No Funcionales**

Adicionalmente, se definieron requerimientos no funcionales que permiten garantizar que la aplicación sea eficiente, segura y compatible con diversos entornos:

* **Interfaz Responsiva:** El diseño de la aplicación se adapta dinámicamente a diferentes tamaños de pantalla y dispositivos, asegurando una experiencia consistente y accesible desde móviles, tablets y computadoras de escritorio.
* **Rendimiento:** La aplicación debe cargar con rapidez y responder ágilmente, incluso en conexiones lentas o intermitentes, para evitar frustración y abandono del usuario.
* **Compatibilidad:** Se garantiza el correcto funcionamiento en los navegadores más utilizados, como Chrome, Firefox, Edge y Safari, maximizando el alcance de la aplicación.
* **Seguridad:** Se aplican prácticas recomendadas para proteger contraseñas, controlar accesos y manejar errores del servidor con mensajes claros y amigables que no comprometan la seguridad.
* **Mantenimiento:** La arquitectura y el código están diseñados para facilitar futuras modificaciones y extensiones, minimizando la dependencia del desarrollador original.
* **Integridad de Datos:** Se previenen duplicados y se mantiene la coherencia referencial en la base de datos para garantizar la calidad y confiabilidad de la información.

**4. Tolerancia a Fallos**

**4.2 Tipos de Fallos Identificados**

Durante la etapa de análisis y desarrollo, se identificaron los siguientes tipos de fallos que la aplicación debe manejar:

* **Fallo de Conexión a Internet**: Usuarios que intentan registrar asistencia sin estar conectados, escenario común en dispositivos móviles.
* **Errores en la Base de Datos**: Problemas al guardar o consultar datos, que pueden deberse a desconexiones, conflictos de escritura o problemas del servidor MySQL.
* **Fallos en Peticiones AJAX**: Errores en la comunicación asíncrona entre frontend y backend, que pueden generar estados inconsistentes o fallos visibles para el usuario.
* **Ingreso Incompleto o Incorrecto de Materias**: Intentos de registrar datos con campos vacíos o con formato inválido.
* **Registro Fuera de Horario**: Intentos de registrar asistencia fuera de los horarios establecidos para cada materia.
* **Recarga Inesperada**: Actualización accidental del navegador que provoca pérdida de datos en formularios abiertos.
* **Errores de Validación**: Fallas al capturar campos incompletos o mal formateados.
* **Sesión Expirada**: Intentos de interactuar con la aplicación cuando la sesión del usuario ha terminado o fue invalidada.

**4.3 Mecanismos de Tolerancia Implementados**

Para mitigar los efectos de estos fallos, se implementaron diversas estrategias:

* **Validaciones en JavaScript y PHP**: Antes de enviar cualquier dato, se valida la integridad y completitud en el cliente y en el servidor, reduciendo errores y ataques.
* **Control de Conexión**: El sistema detecta la falta de conexión a internet y bloquea funciones críticas, notificando al usuario.
* **Manejo de Errores con PDO**: En PHP, se capturan excepciones de la base de datos y se envían mensajes claros en formato JSON al frontend.
* **Gestión de Sesiones**: Se verifica la sesión activa antes de permitir acceso a funciones sensibles, redirigiendo al login si es necesario.
* **Retroalimentación Visual**: Se utilizan alertas y mensajes en pantalla para informar al usuario de errores y situaciones anómalas.
* **Diseño Responsive y Limpieza de Formularios**: Se advierte sobre recargas que pueden perder datos y se previene confusión en la interfaz.

**4.4 Prevención de Defectos**

Se priorizó un desarrollo modular y basado en pruebas exploratorias que ayudaron a detectar errores tempranamente, incorporando validaciones estrictas y controles visuales para minimizar la ocurrencia de defectos.

**4.5 Redundancia y Diversidad**

El sistema usa redundancia funcional (sesiones, validaciones múltiples) y almacenamiento persistente en MySQL para evitar pérdida de datos, garantizando consistencia y disponibilidad.

**5. Discusión**

Comparado con plataformas comerciales como Jibble, Clockify o sistemas institucionales de gestión académica, la aplicación propuesta se distingue por su simplicidad, acceso libre y facilidad de instalación y uso.  
Sistemas comerciales suelen incluir funciones avanzadas para grandes empresas, pero resultan excesivos para contextos educativos simples y pueden requerir licencias o integraciones complejas [5].

El sistema aquí presentado cubre las necesidades esenciales de registro de asistencia, evitando sobrecarga funcional y priorizando la experiencia del usuario final, como sugiere la literatura sobre diseño centrado en el usuario [3].

Entre las limitaciones identificadas están la falta de integración con otras plataformas educativas, la ausencia de reportes avanzados para administración institucional y la autenticación al iniciar sesión usando Google, aspectos que pueden ser abordados en trabajos futuros.

**5. Conclusiones**

El sistema de registro de asistencias presentado demuestra que es posible ofrecer una herramienta digital sencilla, eficiente y accesible para el control académico, evitando la complejidad y sobrecarga de soluciones comerciales existentes.  
Su arquitectura modular y el uso de tecnologías estándar facilitan la adaptación y el mantenimiento. La experiencia del usuario y la seguridad de la información son pilares del diseño, cubriendo las necesidades básicas del entorno educativo digital.  
Como trabajo futuro, se plantea la integración con plataformas de gestión académica, el desarrollo de módulos de análisis avanzado de datos de asistencia y la implementación de autenticación al iniciar sesión usando Google.

# Referencias

1. Xavier Alejandro, L. Código del Sistema de Registro de Asistencia. Disponible en: https://github.com/Alejandro-Leon24/Proyecto-web.git
2. Fernández-García, Á. J., García-Peñalvo, F. J., & Therón, R. (2019). Usability evaluation of learning analytics dashboards: A systematic literature review. Computers & Education, 141, 103647. <https://doi.org/10.1016/j.compedu.2019.103647>
3. Norman, D. A. (2013). The Design of Everyday Things: Revised and Expanded Edition. MIT Press.
4. Nielsen, J. (1994). Usability Engineering. Academic Press.
5. Rashid, A., Rahman, A., & Musa, R. M. (2016). Comparison of Attendance Monitoring Systems: A Review. 2016 3rd International Conference on Computer and Information Sciences (ICCOINS), 2016, 439–444. <https://ieeexplore.ieee.org/document/7733302>
6. Sommerville, I. (2011). Software Engineering (9th ed.). Pearson.
7. Google Developers. (2024). Progressive Web Apps Overview. Disponible en: https://web.dev/progressive-web-apps/
8. Pérez, J., & Salas, M. (2021). Herramientas digitales como apoyo a la autogestión académica en entornos de baja conectividad. Revista Iberoamericana de Educación, 87(2), 45–60.
9. W3C Web Accessibility Initiative. (2018). Web Content Accessibility Guidelines (WCAG) 2.1. Disponible en: https://www.w3.org/TR/WCAG21/