Universidad de la Cuenca del Plata
Autorización Definitiva. Decreto Poder Ejecutivo Nacional N°091/2006

Paradigmas y Lenguajes de Programación III

CARRERA: Ingeniería en Sistemas de Información

MATERIA: Paradigmas y Lenguajes de Programación III COMISIÓN: “U” (única)

PROFESOR: Mgter. Ing. Encina Agustín.

ESTUDIANTE: Gamarra Kiara Barbarella

FECHA: 13-11-2025

Cuadro comparativo de amenazas

Tipo de amenaza	Causas	Consecuencias	Controles preventivos
Ataques cibernéticos	Inyección de código, phishing, fallas de autenticación, explotación de vulnerabilidades OWASP	Robo de datos personales, pérdida de cuentas, caída del sistema, manipulación de información	Validación de entradas, contraseñas encriptadas, HTTPS, control de acceso fuerte, sanitización, pruebas OWASP, autenticación robusta
Desastres naturales	Inundaciones, incendios, tormentas, cortes de energía	Pérdida total del servidor, indisponibilidad del servicio, corrupción de archivos	Backups, servidores en la nube, redundancia, UPS, recuperación ante desastres
Errores Humanos	Contraseñas débiles, borrar archivos por accidente, configuraciones incorrectas, subir código sin testear	Pérdida de información, fallos en producción, vulnerabilidades abiertas, mal funcionamiento de la app	Capacitación, políticas de seguridad, formularios con validación, roles y permisos, automatización en despliegues
Vulnerabilidades técnicas	Bugs, configuraciones por defecto, falta de parches, frameworks desactualizados	Exposición de datos, accesos no autorizados, mal rendimiento, fallos de compatibilidad	Actualizaciones periódicas, mantenimiento, pruebas QA, monitoreo, hardening del servidor

Evaluación de la Aplicación del Trabajo Integrador

Accesibilidad

Se observa un diseño general claro, con formularios e inputs reconocibles. Sin embargo, existen varias limitaciones respecto de las pautas de accesibilidad establecidas por W3C y WCAG. Las imágenes incluidas en las pantallas de inicio de sesión y registro cuentan con textos alternativos poco descriptivos, lo que afecta a lectores de pantalla. Algunos botones no contienen atributos accesibles que comunican su función más allá del texto visible. La navegación mediante teclado no está completamente implementada, especialmente en elementos dinámicos como las tarjetas agregadas por

JavaScript. Los iconos utilizados en diferentes secciones podrían no tener contraste suficiente, lo que perjudica a usuarios con baja visión.

Usabilidad

La aplicación presenta un flujo sencillo que permite al usuario iniciar sesión, acceder al inicio y crear nuevas colecciones. La interfaz es simple y ofrece una curva de aprendizaje baja. No obstante, se identifican dificultades vinculadas a la consistencia y claridad de ciertas interacciones. El formulario de inicio de sesión funciona únicamente de manera local y no ofrece mensajes de error específicos que orienten al usuario frente a datos incorrectos. El proceso de creación de tarjetas depende íntegramente del botón “Listo”, sin confirmación intermedia, lo que puede generar confusión. En algunos casos falta retroalimentación visible que indique que las acciones se han realizado correctamente. Además, no existe una pantalla destinada a visualizar o estudiar cada colección, lo que limita la experiencia de uso.

Seguridad

Actualmente el sistema utiliza almacenamiento local del navegador (localStorage) para guardar usuarios y colecciones. Este mecanismo no es seguro, ya que cualquier dato puede ser manipulado desde la consola del navegador, generando riesgos de integridad y confidencialidad. No existe autenticación real, porque el inicio de sesión sólo guarda el nombre del usuario sin verificación contra una base de datos. Tampoco hay sanitización de entradas, lo que permitiría almacenar código malicioso dentro de un término o definición, provocando posibles ataques de cross-site scripting. La falta de controles de acceso implica que cualquier usuario podría ver o modificar información que no le pertenece una vez que el proyecto se integre con una base de datos real.

Propuestas de Mejora

Se recomienda implementar un sistema de sanitización de datos de entrada con el fin de prevenir ataques basados en la inserción de código malicioso. También se aconseja reemplazar localStorage por un sistema de autenticación real, utilizando PHP y MySQL con funciones seguras como password_hash y password_verify. Para mejorar la accesibilidad, sería conveniente añadir atributos ARIA, controles de foco y textos alternativos descriptivos para todas las imágenes. Asimismo, se recomienda desarrollar una pantalla específica para visualizar y estudiar las colecciones, fortaleciendo de este modo la usabilidad general de la aplicación.

Conclusión

La accesibilidad, la usabilidad y la seguridad forman un conjunto inseparable dentro del desarrollo web moderno. La accesibilidad garantiza que cualquier persona, independientemente de sus capacidades, pueda interactuar con la aplicación; la usabilidad asegura que esa interacción sea fluida, clara y eficiente; y la seguridad protege tanto los datos del usuario como la integridad del sistema. Si alguno de estos elementos falla, la aplicación pierde valor: puede ser inutilizable, excluyente o incluso peligrosa.

Paradigmas y Lenguajes de Programación III

En el análisis de mi proyecto, pude ver cómo decisiones simples —como validar entradas, mejorar el contraste o implementar autenticación adecuada— impactan directamente en la experiencia real del usuario. Entre todos los estándares revisados, considero que el OWASP Top 10 es el más relevante, porque cualquier vulnerabilidad puede comprometer el funcionamiento completo de la aplicación y la confianza del usuario. Las WCAG y la ISO brindan lineamientos muy importantes, pero OWASP aborda riesgos concretos y actuales que afectan a la mayoría de los sistemas web.

Trabajar con estos enfoques integrados me permitió comprender que un buen sistema no es solo “bonito” o “funcional”, sino también inclusivo y seguro.