Paradigmas y Lenguajes de Programación III

CARRERA: Ingeniería en Sistemas de Información

MATERIA: Paradigmas y Lenguajes de Programación III

COMISIÓN: “A”

PROFESOR: Mgter. Ing. Agustín Encina

ESTUDIANTE: Luciana Nadine Rojas

CUADRO COMPARATIVO DE AMENAZAS

Tipo de amenaza	Ejemplos	Causas	Consecuencias	Controles preventivos
Ataques Cibernéticos	- Inyección SQL - XSS (Cross-Site Scripting) - CSRF - DDoS	- Vulnerabilidades en código - Falta de validación de entrada - Configuración insegura	- Robo de datos - Denegación de servicio - Defacement - Pérdida financiera	- Validación de entrada - Consultas preparadas - WAF (Web Application Firewall) - Actualizaciones de seguridad
Desastres Naturales	- Inundaciones - Terremotos - Incendios - Tormentas eléctricas	- Fenómenos climáticos - Eventos geológicos - Factores ambientales	- Daño físico a servidores - Pérdida de datos - Tiempo de inactividad prolongado	- Backup en la nube - Centros de datos redundantes - Plan de recuperación de desastres - UPS y estabilizadores
Errores Humanos	- Configuración incorrecta - Eliminación accidental de datos - Exposición de credenciales - Falta de parches	- Falta de capacitación - Fatiga o distracción - Procedimientos deficientes	- Brechas de seguridad - Corrupción de datos - Tiempo de inactividad no planificado	- Automatización de procesos - Control de acceso basado en roles - Monitoreo y logging - Capacitación continua
Vulnerabilidades Técnicas	- Software desactualizado - Contraseñas débiles - APIs inseguras - Cifrado insuficiente	- Falta de mantenimiento - Diseño deficiente - Pruebas insuficientes	- Explotación por atacantes - Pérdida de confidencialidad - Violación de compliance	- Parches regulares - Escaneo de vulnerabilidades - Code review - Pruebas de penetración

EVALUACIÓN DE MI APLICACIÓN (TODOLIST MVC)

Análisis de Accesibilidad

Cumple:

- Estructura semántica básica en HTML
- Contraste de colores aceptable (#333 sobre #f4f4f9)
- Navegación por teclado funcional en formularios

Necesita Mejora:

- Falta de etiquetas ARIA para elementos dinámicos
- No hay texto alternativo en elementos interactivos
- Focus indicators no visibles en todos los elementos
- No compatible totalmente con lectores de pantalla

Análisis de Usabilidad

Cumple:

- Interfaz limpia y minimalista
- Flujo de tareas intuitivo (añadir → completar → eliminar)
- Feedback visual inmediato al completar tareas
- Consistencia en botones y acciones

Necesita Mejora:

- No hay confirmación para eliminar tarea
- Falta de breadcrumbs o indicador de ubicación
- No hay búsqueda o filtrado de tareas
- Mensajes de error genéricos

Análisis de Seguridad

Cumple:

- Uso de consultas preparadas en TareaModel.php

- Validación básica de entrada en TareaController.php
- Escapado de output con htmlspecialchars()

Necesita Mejora:

- No hay protección CSRF en formularios
- Validación de entrada insuficiente
- No hay rate limiting para prevenir spam
- Mensajes de error podrían revelar información sensible

MEJORAS PROPUESTAS

Mejora 1: Implementación de Accesibilidad WCAG 2.1 AA

Archivos a modificar: task_list.php, style.css

Html:

```
<!-- Mejoras en task_list.php -->
<main role="main">
    <h1 id="main-title">To-Do List MVC (PHP)</h1>

    <form method="POST" action="index.php?action=crear"
          aria-labelledby="main-title" class="task-form">
        <label for="task-input" class="sr-only">Nueva tarea</label>
        <input type="text" id="task-input" name="descripcion"
               placeholder="Escribe tu nueva tarea..." required
               aria-required="true">
        <button type="submit" aria-label="Añadir nueva tarea">Añadir</button>
    </form>
```

```
    <div role="status" aria-live="polite" id="status-message"></div>
</main>
```

Css:

```
/* Mejoras en style.css */
.sr-only {
    position: absolute;
    width: 1px;
    height: 1px;
    padding: 0;
    margin: -1px;
    overflow: hidden;
    clip: rect(0, 0, 0, 0);
    border: 0;
}

button:focus, input:focus, a:focus {
    outline: 3px solid #4a90e2;
    outline-offset: 2px;
}

.task-list li:focus-within {
    box-shadow: 0 0 0 3px rgba(74, 144, 226, 0.5);
}
```

Mejora 2: Fortalecimiento de Seguridad OWASP

Archivos a modificar: TareaController.php, index.php

Php:

```
<?php
// En index.php - agregar al inicio
session_start();

// Generar y almacenar token CSRF
if (empty($_SESSION['csrf_token'])) {
    $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
}

// En TareaController.php - método crear()
public function crear() {
    // Validar CSRF token
    if ($_SERVER['REQUEST_METHOD'] == 'POST') {
        if (!isset($_POST['csrf_token']) ||
            $_POST['csrf_token'] !== $_SESSION['csrf_token']) {
            die('Token CSRF inválido');
        }

        // Validación robusta de entrada
        $descripcion = trim($_POST['descripcion'] ?? '');
```

```
        if (empty($descripcion)) {
            $_SESSION['error'] = "La descripción no puede estar vacía";
        } elseif (strlen($descripcion) < 3) {
            $_SESSION['error'] = "La descripción debe tener al menos 3 caracteres";
        } elseif (strlen($descripcion) > 255) {
            $_SESSION['error'] = "La descripción no puede exceder 255 caracteres";
        } else {
            // Sanitización adicional
            $descripcion = filter_var($descripcion, FILTER_SANITIZE_STRING);
            $descripcion = htmlspecialchars($descripcion, ENT_QUOTES, 'UTF-8');

            $this->model->crearTarea($descripcion);
            $_SESSION['success'] = "Tarea creada correctamente";
        }
    }
    header("Location: index.php");
    exit();
}
```

Html:

```
<!-- En task_list.php - agregar campo hidden -->
<form method="POST" action="index.php?action=crear" class="task-form">
    <input type="hidden" name="csrf_token" value="<?= $_SESSION['csrf_token'] ?>">
```

```
    <!-- resto del formulario -->
</form>
```

Conclusión:

La accesibilidad, usabilidad y seguridad son tres pilares interdependientes que determinan la calidad integral de una aplicación web. La accesibilidad garantiza que todos los usuarios, independientemente de sus capacidades, puedan interactuar con el sistema. La usabilidad asegura que esta interacción sea intuitiva, eficiente y satisfactoria. La seguridad protege tanto a los usuarios como a la aplicación de amenazas internas y externas.

Estos tres aspectos se relacionan de manera sinérgica: una aplicación segura pero inaccesible excluye usuarios; una aplicación accesible pero insegura pone en riesgo datos sensibles; y una aplicación usable pero no accesible limita su alcance. Por ejemplo, las validaciones de seguridad mejoran la usabilidad al prevenir errores, y una buena usabilidad reduce errores humanos que podrían comprometer la seguridad.

Entre las normas analizadas, considero que OWASP Top 10 es la más relevante para el desarrollo actual, ya que aborda vulnerabilidades concretas y actualizadas que afectan a las aplicaciones web modernas. Su enfoque práctico y específico proporciona guías accionables inmediatas para desarrolladores. Sin embargo, la ISO 9126 (sucesora ISO 25000) ofrece una visión holística invaluable para evaluar la calidad del software desde múltiples dimensiones.

La implementación balanceada de estos estándares, adaptada al contexto específico de cada proyecto, es esencial para crear aplicaciones web robustas, inclusivas y confiables que entreguen valor real a los usuarios finales mientras protegen sus datos y privacidad.