**DIGITAL NAO**

**Nombre completo**

Edwin Jair Sánchez Cevallos

**NAO ID**

3078

**Nombre de la trayectoria**

Consultor de Ciberseguridad

**Título del Reto**

Protocolos de seguridad con pentesting y criptografía

**Fecha**

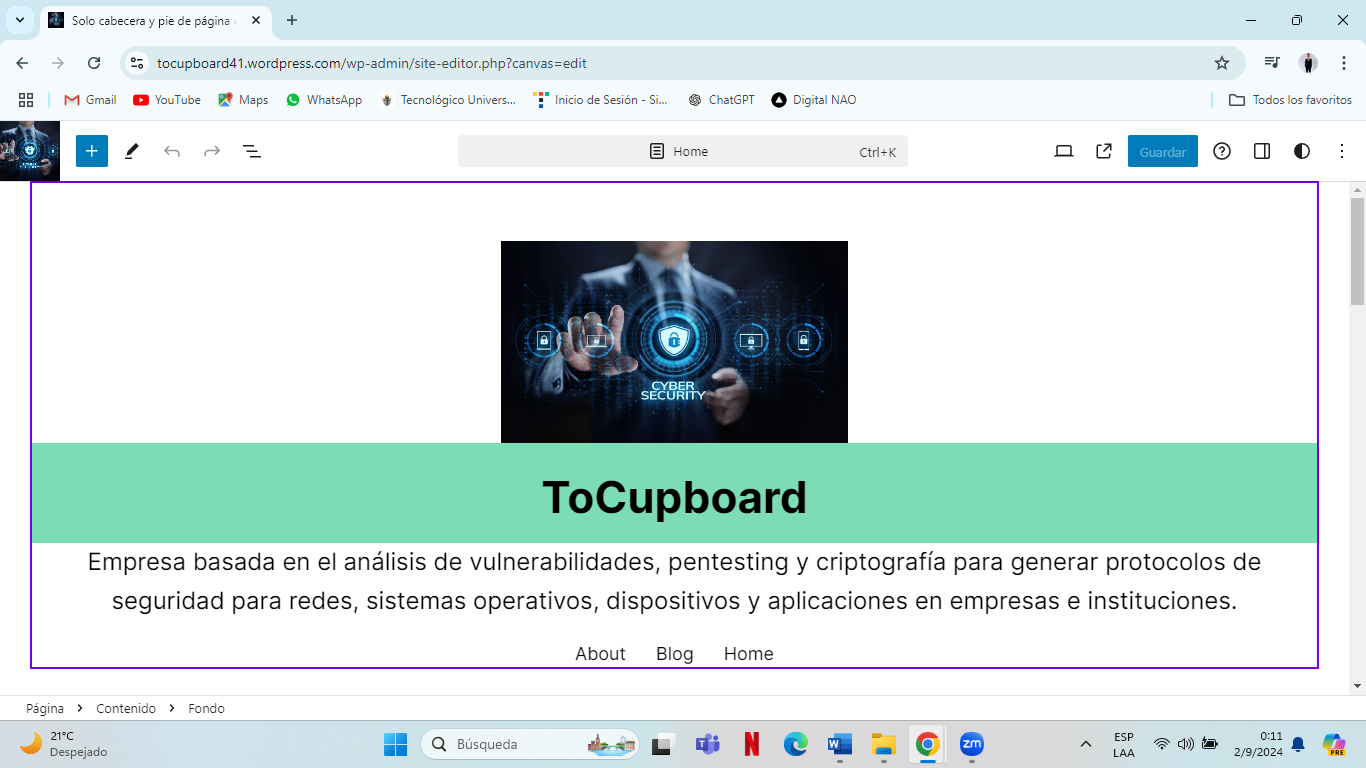
2 de septiembre del 2024

**SPRINT 2**

**1. Desarrolla una página web con WordPress, que incluya llamadas a una API y una simulación de una**[**pasarela de pagos**](https://github.com/FabianCristancho/pasarela-de-pagos)**, aplicando el modelo DevSecOps.**

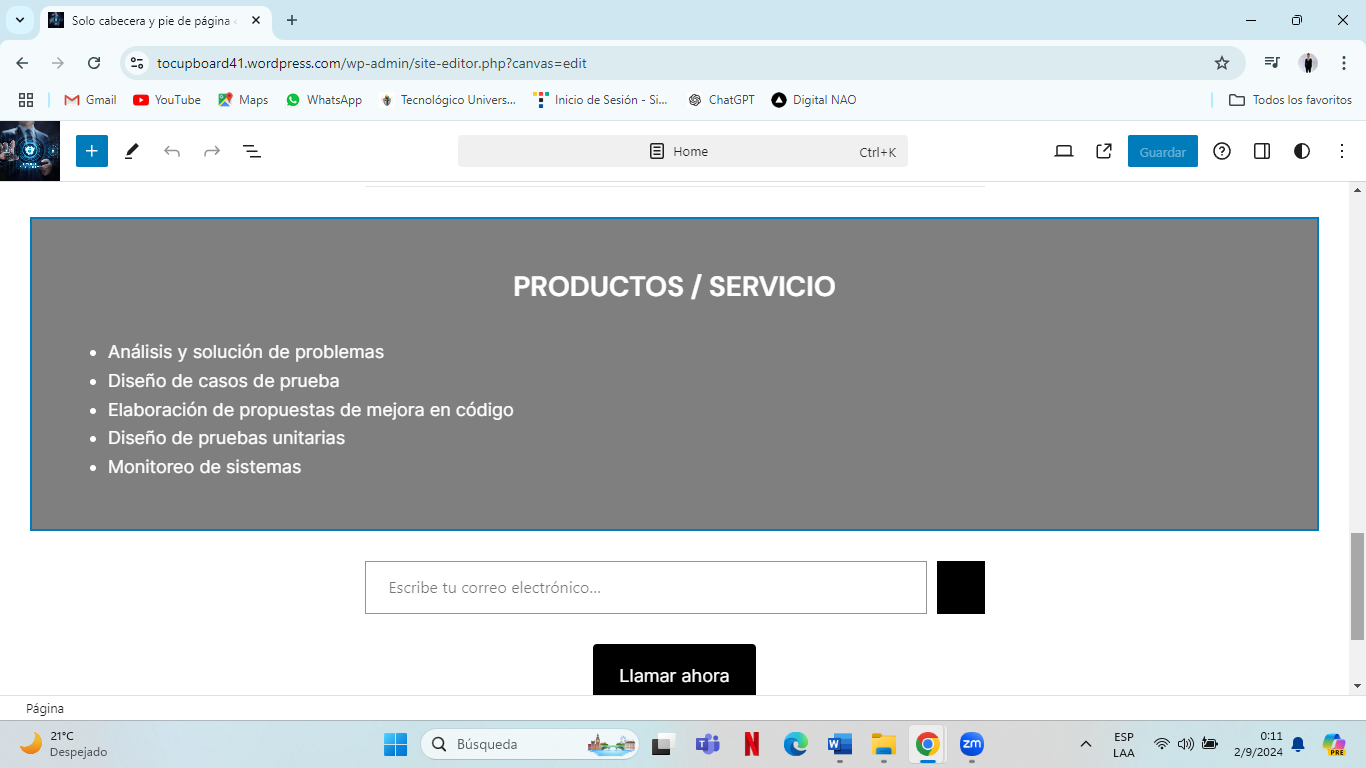
**Página Web en WordPress:**

Tema y Diseño: Utiliza un tema apropiado para la empresa ToCupboard, con un diseño profesional y responsive.

****Contenido Básico: Incluye páginas como Inicio, Sobre Nosotros, Productos/Servicios, Contacto.

**Interfaz de usuario gráfica

Descripción generada automáticamente**

****

**Seguridad en el Desarrollo: Asegúrate de que el sitio web cumpla con las mejores prácticas de seguridad desde el diseño, incluyendo:**

**Actualización de WordPress y sus plugins**: Mantén siempre actualizados tanto el core de WordPress como los plugins y temas. Usa un plugin como "Easy Updates Manager" para gestionar actualizaciones automáticas o revisa manualmente el panel de actualizaciones regularmente.

**Uso de plugins de seguridad.**

* Wordfence: Protección en tiempo real, firewall y escaneo de malware.
* iThemes Security: Refuerza la seguridad con más de 30 maneras de proteger tu sitio.
* WP fail2ban: Registra los intentos de login fallidos para prevenir ataques de fuerza bruta.

**Configuración de HTTPS.**

* Certificado SSL: Asegúrate de instalar un certificado SSL en el servidor para habilitar HTTPS. Esto cifrará las comunicaciones entre el servidor y los usuarios.
* Solución: Puedes usar "Let’s Encrypt" para obtener un certificado gratuito. Instala un plugin como "Really Simple SSL" para forzar todas las conexiones a HTTPS.

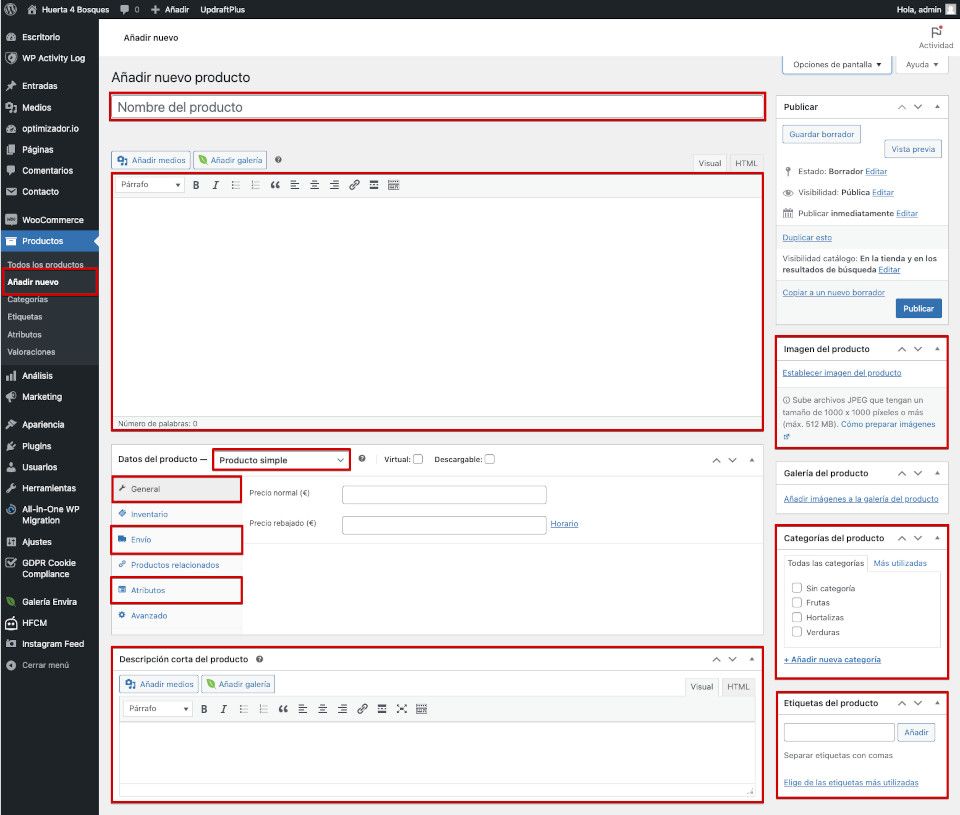
**Llamadas a una API:**

**Integración de API: Implementa llamadas a una API externa o propia, asegurando la correcta integración con el sitio web.**

* Identifica la API que deseas utilizar (externa o propia).
* Asegúrate de que la API esté documentada y tenga soporte para autenticación.
* Usa un plugin como WP REST API si estás trabajando con una API propia.

**Ejemplos de Llamadas: Incluye al menos dos ejemplos de llamadas API (por ejemplo, obtención de datos de productos, envío de formularios).**

1. **Obtención de datos de productos**

* Instalar un Plugin de E-commerce
* Si estás utilizando WooCommerce, asegúrate de que el plugin esté instalado y activado.
* Configura tus productos en WooCommerce, incluyendo nombre, descripción, precio, imágenes, etc.
* Uso de Shortcodes o Funciones PHP
* Para mostrar los productos en páginas específicas, puedes utilizar shortcodes de WooCommerce.

**2. Envíos de formularios**

1. Instalar un Plugin de Formularios

* Plugins populares: Contact Form 7, WPForms, Gravity Forms.
* Instala y activa el plugin de formularios de tu elección.

1. Crear un Formulario

* Crea un formulario usando el plugin instalado.
* Configura los campos del formulario según tus necesidades (nombre, correo electrónico, mensaje, etc.).

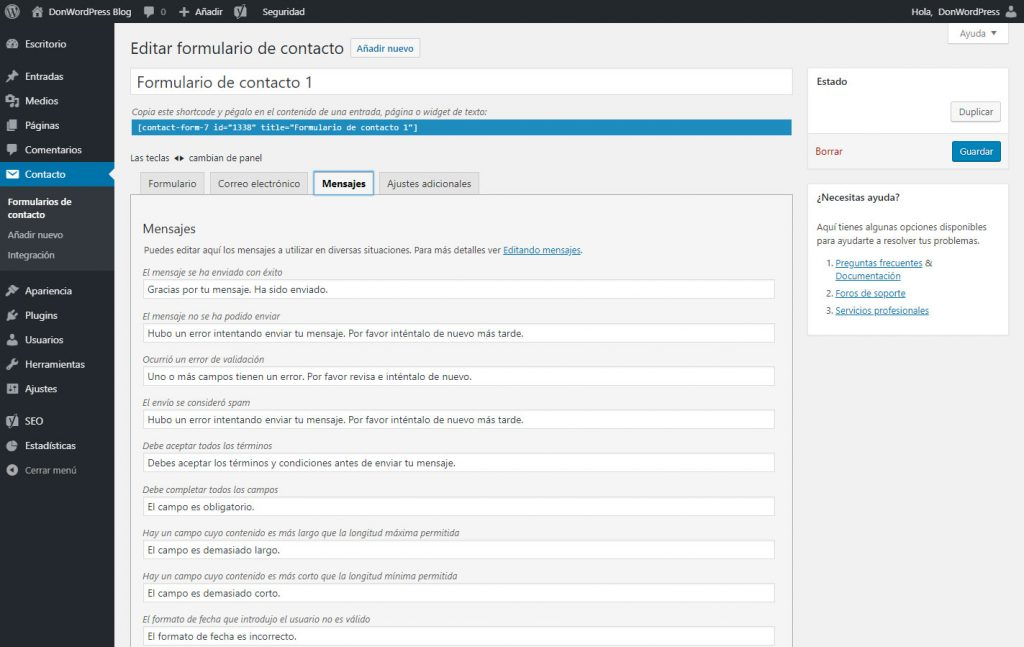
1. Mostrar el Formulario en una Página

* Inserta el formulario en una página o entrada de WordPress usando el shortcode generado por el plugin.

1. Procesamiento del Formulario

* Configura las opciones de procesamiento, como el correo electrónico de destino para recibir las respuestas, mensajes de confirmación, redirección a una página específica después del envío, etc.
* Algunos plugins permiten añadir lógica personalizada o integraciones con otros servicios (como CRMs, bases de datos, etc.).

1. Validación y Seguridad

* Asegúrate de que el formulario incluya validación de entrada para evitar datos incorrectos.
* ****Implementa medidas de seguridad como reCAPTCHA o honeypots para evitar envíos de spam.

**Seguridad en las Llamadas: Asegura que las llamadas a la API sean seguras, utilizando métodos de autenticación y autorización adecuados.**

**Autenticación y Autorización:**

* Tokens Bearer: Como en los ejemplos anteriores, puedes utilizar un token de acceso en el encabezado de la solicitud para autenticación.
* OAuth 2.0: Si tu API lo soporta, considera implementar OAuth 2.0 para mejorar la seguridad.

**Uso de HTTPS:**

* Todas las llamadas a la API deben realizarse a través de conexiones HTTPS para proteger los datos en tránsito.

**Limitación de Tasa de Solicitudes:**

* Implementa mecanismos para limitar la tasa de solicitudes a la API, protegiendo así tu servidor de ataques de denegación de servicio (DoS).

**Simulación de una Pasarela de Pagos:**

**Proceso de Pago: Implementa una simulación del proceso de pago, desde la selección del producto hasta la confirmación del pago.**

1.Selección de Producto

* Los usuarios navegarán por tu tienda, seleccionarán productos y los añadirán al carrito.
* Asegúrate de que el carrito y las páginas de pago estén correctamente configuradas.

2. Proceder al Pago

* Una vez que el usuario esté listo para pagar, accederá a la página de checkout.
* Aquí se presentarán las opciones de pago que configuraste (Stripe o PayPal en modo de prueba).

3. Introducción de Datos de Pago

* En el checkout, los usuarios introducirán datos de pago simulados.
* Stripe: Puedes usar tarjetas de prueba proporcionadas por Stripe, como 4242 4242 4242 4242 con cualquier CVC y fecha futura.
* PayPal: Los usuarios serán redirigidos a PayPal Sandbox para completar el pago con una cuenta de prueba.

4. Confirmación del Pago

* Después de completar el pago, el usuario será redirigido a la página de confirmación de pedido en tu tienda.
* Aquí se mostrará un mensaje de confirmación del pago y detalles del pedido.

**Seguridad en el Pago: Asegura que la simulación siga las mejores prácticas de seguridad en pagos, como:**

**Uso de Tokens de Pago**

* Stripe y PayPal usan tokens de pago en lugar de almacenar datos de tarjetas directamente. Esto asegura que los datos de pago nunca se transmiten ni almacenan en tu servidor.
* Stripe genera un token de pago cada vez que se procesa una tarjeta, y PayPal maneja la transacción en su servidor.

**Validación de Datos del Cliente**

* Implementa validación en el front-end y back-end para asegurarte de que los datos del cliente sean correctos.
* Utiliza plugins de seguridad para proteger formularios de pagos y datos sensibles.
* Validación de tarjetas de crédito: Stripe y PayPal hacen esto automáticamente, pero puedes añadir validación adicional.

**Simulación de Entornos Seguros**

* HTTPS: Asegúrate de que tu sitio esté configurado con HTTPS para proteger la transmisión de datos.
* Modo de Prueba: Usa entornos de prueba como el Sandbox de PayPal o el modo de prueba de Stripe para asegurar que no se procesen pagos reales.

**Aplicación del Modelo DevSecOps:**

**Integración del Modelo: Describe cómo se ha aplicado el modelo DevSecOps en la implementación de la página web.**

En la implementación de la página web de ToCupboard, el modelo DevSecOps se ha integrado para asegurar que la seguridad esté presente en todas las etapas del ciclo de vida del desarrollo de software (SDLC). Este enfoque combina las prácticas de desarrollo (Dev), operaciones (Ops) y seguridad (Sec) en un proceso continuo que abarca desde la planificación y codificación hasta el despliegue y monitoreo. Se utilizaron plugins y herramientas de seguridad en los entornos de desarrollo integrados (IDE) para detectar vulnerabilidades en el código durante la fase de desarrollo.

**Prácticas de Seguridad: Documenta las prácticas de seguridad implementadas, como:**

1. **Integración continua y despliegue continuo (CI/CD).**

* Pipeline CI/CD Seguro: Se configuró un pipeline CI/CD utilizando herramientas como Jenkins y GitLab CI. Este pipeline asegura que cada cambio en el código pase por pruebas automáticas de seguridad antes de ser desplegado en producción.
* Escaneo de Vulnerabilidades: Cada commit se escanea automáticamente en busca de vulnerabilidades de seguridad usando herramientas como SonarQube o OWASP Dependency-Check. Esto incluye análisis estático del código y escaneo de dependencias.
* Despliegue Automatizado con Contenedores Seguros: La aplicación se empaqueta en contenedores Docker que son revisados y escaneados en busca de vulnerabilidades antes del despliegue en entornos de prueba y producción.

1. **Pruebas automatizadas de seguridad.**

* Pruebas de Seguridad Automatizadas: Se han integrado pruebas automatizadas de seguridad como parte del pipeline CI/CD, utilizando herramientas como Zaproxy o Burp Suite para realizar pruebas de penetración automatizadas y análisis de seguridad.
* Pruebas de Seguridad Dinámicas (DAST): Se implementaron pruebas de seguridad dinámicas que simulan ataques en la aplicación web para identificar posibles vulnerabilidades durante su ejecución.
* Análisis de Comportamiento del Código (SAST): El análisis estático del código se realiza automáticamente para detectar posibles vulnerabilidades en el código fuente antes de que sea desplegado.

1. **Monitoreo continuo de vulnerabilidades**

* Monitoreo en Tiempo Real: Se utilizan herramientas de monitoreo como Splunk y ELK Stack para realizar un monitoreo continuo de la aplicación, detectando y alertando sobre posibles amenazas o anomalías en tiempo real.
* Gestión de Parches y Actualizaciones: Se implementa un proceso automatizado para aplicar parches de seguridad y actualizar componentes vulnerables tan pronto como se identifican.
* Auditorías Regulares de Seguridad: Se programan auditorías de seguridad regulares para revisar la infraestructura y asegurar que se cumplen todas las políticas y mejores prácticas de seguridad.