🧾 Checklist básica de seguridad para desarrolladores web

🔍 1. Validación de entradas y salidas

- [ ] Todas las **entradas de usuario** (formularios, parámetros URL, JSON, etc.) se **validan y sanitizan** antes de procesarse.
- [ ] No se usan **concatenaciones directas en consultas SQL** (usar **prepared statements** o **ORMs**).
- [ ] Se aplican **restricciones de longitud, tipo y formato** a los datos (por ejemplo: emails, IDs numéricos, etc.).
- [ ] Las **respuestas del servidor** escapan correctamente caracteres HTML para evitar **XSS reflejado o almacenado**.

🗝️ 2. Manejo de autenticación y contraseñas

- [ ] Las contraseñas se **hashean con bcrypt, Argon2 o scrypt** (nunca guardarlas en texto plano).
- [ ] Los tokens o sesiones tienen **tiempo de expiración razonable**.
- [ ] No hay **tokens, contraseñas ni claves API** dentro del código o archivos públicos (usar `.env`).
- [ ] Se fuerza el **uso de HTTPS** para cualquier login o formulario sensible.

🧱 3. Seguridad del servidor y configuración básica

- [ ] Los **encabezados HTTP** de seguridad están configurados (CSP, HSTS, X-Frame-Options, X-Content-Type-Options).
- [ ] El servidor **no expone versión del framework o lenguaje** (por ejemplo, “X-Powered-By: Express”).

- [ ] Los **directorios y archivos sensibles** (`.env`, `.git`, `/config/`, etc.) **no son accesibles públicamente**.
- [ ] Las **cookies** de sesión tienen `HttpOnly`, `Secure` y `SameSite` activados.

📦 4. Dependencias y librerías

- [ ] Todas las dependencias están **actualizadas** a versiones estables.
- [ ] Se ha ejecutado un **auditor de vulnerabilidades** (`npm audit`, `pip-audit`, `composer audit`, etc.).
- [ ] No se usan librerías sin mantenimiento o con baja reputación.
- [ ] ---

🧾 5. Manejo de errores y logs

- [ ] Los **mensajes de error** no revelan información sensible (rutas, consultas, variables, etc.).
- [ ] Los **logs** no guardan contraseñas, tokens ni datos personales.
- [ ] Se usa un sistema de logs estructurado y, si es posible, **rotación o limpieza periódica**.
- [ ] En producción, se desactiva el **modo debug** o `stack traces`.

🧪 6. Pruebas básicas de seguridad antes de entregar

- [ ] Revisar manualmente si los **formularios** permiten insertar scripts (`<script>`) → **si sí, hay XSS**.
- [ ] Probar si los campos permiten `' OR '1'='1` → **si sí, hay SQLi**.
- [ ] Escanear con **OWASP ZAP** o **Burp Suite Community** para detectar fallos comunes.
- [ ] Revisar los resultados y corregir cualquier hallazgo antes de subir o entregar el proyecto.

✅ 7. Buenas prácticas generales

- [ ] El repositorio **no contiene archivos sensibles** antes de subirlo (usar `.gitignore` correctamente).
- [ ] Se revisó que el proyecto **funciona igual en modo producción** que en desarrollo.
- [ ] Se documentaron las **medidas de seguridad aplicadas** en el README o documentación técnica.
- [ ] Todo el código fue **revisado al menos una vez por otra persona o con una IA**.