

Comando de implementación para Evaluador Punto Seguro

El siguiente comando describe de forma detallada cómo construir la herramienta web “**Evaluador Punto Seguro**”, una aplicación interactiva (en **React + Tailwind CSS**) para evaluar el riesgo de robo en viviendas. Integra principios profesionales de seguridad (CPTED, ISO 22341, criminología ambiental) y datos estadísticos reales para generar un **puntaje de riesgo de 0 a 100**, con interfaz moderna, responsive (mobile-first) y totalmente en castellano. El objetivo es que Codex genere el código de la interfaz, la lógica de cálculo y la presentación de resultados siguiendo estas especificaciones.

Fundamentos profesionales de la evaluación de riesgo

- **CPTED e ISO 22341:** La herramienta se basa en *Crime Prevention Through Environmental Design (CPTED)* y en las guías de la norma **ISO 22341:2021**. Esto implica evaluar cómo el entorno físico influye en la seguridad y cómo el diseño puede reducir oportunidades de delito ¹ ². Por ejemplo, una vivienda bien iluminada, con vigilancia natural (ojos de vecinos en la calle) y control de accesos reduce significativamente las posibilidades de intrusión ². La ISO 22341 promueve una comprensión común de estos principios preventivos para crear entornos residenciales más seguros y mejorar la calidad de vida ¹.
- **Criminología ambiental:** Se incorporan teorías de criminología ambiental y situacional. El modelo asume que los ladrones son *agentes racionales* que analizan riesgo/beneficio antes de atacar ³. Por ello, evaluamos factores como la **oportunidad** (ej. cuándo la vivienda está desocupada) y la **atractividad/vulnerabilidad** del objetivo (ej. accesos fáciles, valor esperado del botín) ⁴. *Genoma del Robo* (método GDR) sugiere clasificar indicadores de riesgo en dimensiones de Oportunidad, Atractividad y Vulnerabilidad ⁴, y nuestra herramienta sigue un enfoque similar para cubrir todos estos aspectos.
- **Estudios y datos reales:** Las ponderaciones de cada factor se respaldan en datos estadísticos y estudios. Por ejemplo, se sabe que la mayoría de robos residenciales ocurren **durante el día** (no de noche) aprovechando casas vacías entre semana ⁵. También está comprobado que la **ausencia de vigilancia** (ej. pocos vecinos o entorno despoblado) aumenta la confianza del delincuente y el riesgo de robo ⁶. Por el contrario, medidas de *seguridad activa* como **alarmas** y **cámaras** tienen un fuerte efecto disuasorio: muchos intrusos abandonan el intento si suena una alarma o ven cámaras ⁷, de hecho la mayoría de delincuentes evita por completo viviendas con alarma y desiste al encontrar un sistema activo ⁸. Estos hallazgos reales se traducen en la lógica de puntuación (sumando riesgo cuando faltan medidas, restando cuando existen protecciones).

Estructura y preguntas del formulario de evaluación

El formulario se divide en **5 secciones temáticas** con un total de ~12-15 preguntas breves en lenguaje claro. Cada sección se presenta con un diseño intuitivo, usando controles de entrada simples (checkbox,

select, sliders, botones tipo toggle) junto con íconos o ilustraciones que ayudan a entender visualmente cada pregunta. Las secciones y ejemplos de preguntas son:

- **Entorno:** Evalúa las características del entorno alrededor de la vivienda.

- *Ejemplos de preguntas:*

- “¿Cómo es la **iluminación** de la calle o zona por la noche?” (Opciones: Muy iluminada / Suficiente / Poca o nula iluminación).
- “¿El **barrio** cuenta con vigilancia (vecinal o seguridad privada)?” (Sí / No / No lo sé).
- “¿Hay alta **afluencia de personas** o la zona está aislada normalmente?” (Barrio concurrido / Poco tránsito / Zona aislada).

UI: Podría usar iconos de farola para iluminación, un escudo o cámara para vigilancia, y gráficos de casas cercanas para densidad vecinal. Preguntas clave sobre entorno reflejan principios CPTED de *vigilancia natural* e *iluminación*: un entorno bien vigilado e iluminado disuade a la mayoría de delincuentes ⁹.

- **Tipo de inmueble:** Recopila información básica del inmueble y su ubicación física.

- *Ejemplos de preguntas:*

- “¿El inmueble es una **casa independiente**, un adosado, o un piso en edificio?” (Selector de tipo).
- “¿En qué **planta** se encuentra?” (Baja, intermedia, ático, etc.).
- “¿Tiene colindantes (paredes compartidas) o está **aislado**?” (Aislado / Comparte medianeras).

UI: Aquí se pueden mostrar siluetas de edificio, chalet, etc. Estas preguntas importan porque, por ejemplo, las viviendas en planta baja o aisladas suelen ser más accesibles para intrusos (más riesgo) que un piso alto dentro de un edificio con vecinos.

- **Accesos:** Indaga sobre puertas, ventanas y demás puntos de entrada a la vivienda.

- *Ejemplos de preguntas:*

- “Puerta principal: ¿Dispone de **puerta de seguridad** o blindada?” (Sí / No / No seguro).
- “¿Tiene **rejas o barrotes** en las ventanas accesibles desde el exterior?” (Todas / Algunas / Ninguna).
- “¿Existen otros **puntos de acceso** (balcones, garaje, patio) fáciles de escalar o forzar?” (Sí / No, o lista de checkbox con “balcón”, “ventana a calle”, etc.).

UI: Uso de iconos como una puerta, ventana, garaje. Se incluyen tooltips con aclaraciones (por ejemplo, definir qué cuenta como ventana “accesible”). **Nota:** Las rejas en ventanas son un elemento recomendado de seguridad física; actúan como barrera y **reducen el riesgo** de intrusión ¹⁰, por lo que si el usuario indica que *no* tiene rejas en ventanas vulnerables, el sistema marcará un riesgo mayor. Del mismo modo, puertas reforzadas aumentan la resistencia y reducen la probabilidad de entrada forzada.

- **Rutinas:** Pregunta sobre los hábitos de ocupación de la vivienda, horarios, etc.

- *Ejemplos de preguntas:*

- “¿Cuántas **horas al día** suele estar la vivienda desocupada entre semana?” (Rango numérico o opciones: <4h, 4–8h, >8h).

- “¿La vivienda queda **vacía por largos periodos** (vacaciones prolongadas) regularmente?” (Casi nunca / Algunos fines de semana / Vacaciones >1 semana seguidas).
- “¿Existen **mascotas o personas** que se queden en casa cuando usted no está?” (Nadie / Mascota / Canguro/empleado / Familiar).

UI: Podría usar un control de rango deslizante para horas desocupadas, y toggles para presencia de mascotas/personas. Esta sección captura la *oportunidad* que tendría un ladrón: a mayor tiempo sin ocupantes, mayor riesgo ⁵. Por ejemplo, si la casa suele estar vacía en horario laboral y además los vecinos colindantes también (p. ej. viviendas de veraneo desocupadas), la oportunidad de robo crece significativamente. La herramienta considera esas combinaciones para el cálculo (p. ej. muchas horas desocupada + entorno poco vigilado = alto riesgo).

• **Seguridad Activa:** Pregunta sobre las medidas de seguridad instaladas y hábitos de protección.

• *Ejemplos de preguntas:*

- “¿Cuenta con **sistema de alarma** conectado a central receptora?” (Sí, monitoreada / Sí, local (sirena) / No tiene).
- “¿Tiene **cámaras de seguridad** visibles en la propiedad?” (Sí, exteriores / Sí, interiores solo / No).
- “¿Dispone de **iluminación automática** o sensores de movimiento exteriores?” (Sí / No).
- “¿Cuál es el **hábito al salir** de casa? (Cerrar bien puertas y ventanas / A veces queda algo abierto / Otra persona siempre queda en casa, etc.)”.

UI: Iconos de alarma, cámara, bombilla, candado, etc. Aquí es importante el lenguaje de las preguntas para generar confianza: por ejemplo, si el usuario no tiene alarma, la pregunta podría incluir un tip amigable: “*¿Sabías que las alarmas conectadas reducen significativamente el riesgo de robo?*”. Las alarmas y cámaras incrementan la probabilidad de detección; se reflejará en el puntaje restando riesgo si existen. Estudios respaldan que un intruso suele desistir si detecta una alarma activa ⁸ o cámaras visibles, especialmente combinadas con buena iluminación ¹¹.

Detalles de UI comunes: Cada sección del formulario estará claramente separada, con un título breve y descripción si necesaria. Se utilizará un lenguaje **formal pero comprensible**, evitando jerga técnica; por ejemplo, decir “ventanas con rejas” en lugar de “protecciones perimetrales”. Los textos deben inspirar confianza y rigor, por lo que se pueden incluir pequeñas notas del tipo “(Basado en estándares profesionales CPTED e ISO 22341)” para respaldar la credibilidad. Al avanzar entre secciones, se proporciona retroalimentación visual (por ejemplo, un *check* o cambio de color en la barra de progreso del formulario) indicando que esa parte está completa.

Lógica de cálculo de la puntuación (0–100)

Cada respuesta del formulario alimenta un **algoritmo de scoring** que calcula un índice de riesgo entre **0 (riesgo nulo)** y **100 (riesgo muy alto)**. La puntuación se compone ponderando profesionalmente los factores de riesgo, de manera que refleje el impacto real de cada elemento según estudios criminológicos y datos.

• **Ponderaciones basadas en riesgo real:** A cada respuesta se le asigna un valor numérico (positivo o negativo) según aumente o disminuya el riesgo. Estos valores están calibrados con criterio profesional. Por ejemplo: **+25 puntos** si el inmueble tiene *ventanas sin rejas* accesibles (vulnerabilidad grave, ya que las rejas actúan como elemento disuasorio físico ¹⁰), **+10** si la iluminación exterior es deficiente (facilita acciones encubiertas), **-15 puntos** si *existe una alarma conectada* a central (medida que **disuade** a la mayoría de ladrones ⁸), **-10** si hay cámaras

visibles en el exterior, -5 si la puerta es blindada de alta seguridad, etc. De este modo, factores que aumentan la *atractividad* del objetivo para el delincuente (como falta de protecciones, accesos débiles) sumarán riesgo, mientras que factores que incrementan la *resistencia o detección* (alarmas, cámaras, vigilancia) restarán riesgo neto. El algoritmo también puede contemplar interacciones: p. ej., la ausencia de alarma **y** de vecinos vigilantes simultáneamente puede sumar un pequeño extra, reflejando que el sistema de alarma suple en parte la falta de vigilancia humana.





- **Escalamiento profesional del puntaje:** No se trata de una suma simple de “sí = 1 punto, no = 0”, sino de una suma ponderada donde ciertos factores críticos pesan más. Se pueden utilizar rangos predefinidos de riesgo para clasificar el resultado: por ejemplo, en evaluaciones profesionales se suele considerar <40 puntos como *riesgo muy bajo*, 40–60 bajo, ~60–75 medio, 75–88 alto, >88 muy alto ¹². Nuestra herramienta adoptará umbrales similares para colorear la visualización (verde, amarillo, rojo) y dar contexto al usuario. El cálculo final asegurará que un usuario que responde todas las preguntas indicando máxima seguridad (ej. entorno vigilado, puertas y ventanas reforzadas, alarma, hábitos prudentes) obtenga un puntaje bajo cercano a 0 (riesgo mínimo), mientras que alguien con múltiples vulnerabilidades (casa aislada, sin iluminación, sin alarma, largas ausencias) obtenga un puntaje alto acercándose a 100 (riesgo elevado).
- **Transparencia y trazabilidad:** El código generado debe estructurar la lógica de puntuación de forma clara (por ejemplo, en una función que suma/subtrae puntos por cada respuesta). Esto facilita futuras actualizaciones en las ponderaciones conforme aparezcan nuevos estudios o recomendaciones. Además, permite explicar al usuario el resultado: junto con el score numérico, el sistema identificará qué respuestas/factores **contribuyeron más** al riesgo total, para mostrarlos en el informe final (ver siguiente sección).

Presentación de resultados de la evaluación

Ejemplo de indicador visual tipo dial para el nivel de riesgo (bajo en verde, medio en amarillo, alto en rojo). La herramienta mostraría un dial de este estilo junto al puntaje numérico obtenido.

Al finalizar el cuestionario, la herramienta presenta al usuario una **pantalla de resultados** atractiva y fácil de interpretar. Los componentes clave de esta UI de resultado son:

- **Indicador gráfico de riesgo:** Un medidor estilo **dial/velocímetro** o una **barra de progreso** horizontal, que muestre el nivel de riesgo en una escala de colores desde verde (seguro) hasta rojo (peligroso). Este indicador debe animarse suavemente hasta la posición del puntaje obtenido (por ejemplo, la aguja del dial girando desde 0 hasta el valor final con una animación de 2 segundos). El dial incluye secciones de color: verde para 0–40 (riesgo bajo), amarillo para medio, rojo para alto, alineado con los rangos mencionados ¹². Se debe asegurar también que haya un texto que indique la categoría (por ejemplo “Riesgo Bajo”/“Medio”/“Alto”) para accesibilidad y claridad.
- **Puntaje numérico grande:** En el centro o al lado del dial se muestra el número de puntuación de 0 a 100, enfatizado tipográficamente. Por ejemplo: **“72/100”**. Junto al número, se puede agregar una breve etiqueta textual como “Riesgo Alto” para reforzar el mensaje del color. El estilo visual será moderno y limpio: tipografía sans-serif clara, con suficiente contraste. Este número también podría tener un efecto de conteo ascendente al revelarse (e.g., un contador animado que sube de 0 al 72).

- **Explicación de factores principales:** Debajo del resultado principal, se presenta un breve informe en lenguaje muy **claro y coloquial**, destacando los **3 a 5 factores** que más influyeron en el resultado. Esto se puede mostrar como una lista o como tarjetas:
- Cada factor listado incluirá una frase explicativa y quizás un pequeño icono. Por ejemplo: “*Ventanas sin rejas: Hemos detectado que tu vivienda tiene ventanas sin protección de rejas. Esto aumenta mucho el riesgo, ya que facilita la entrada de intrusos. Te recomendamos instalar rejas de seguridad o cristales reforzados en esas ventanas ¹³.*” Otro ejemplo: “ *Alarma conectada: Buen trabajo, dispones de una alarma monitoreada. Esto reduce el riesgo considerablemente, muchos ladrones evitan viviendas con alarma activa ⁸. Mantén siempre activo el sistema cuando la casa esté vacía.*”. Cada explicación utiliza iconografía de semáforo ( riesgo,  precaución,  fortaleza) para que el usuario identifique cuáles factores fueron negativos y cuáles positivos en su evaluación. Tras cada factor negativo, se ofrece una **recomendación concreta** de mejora (ej.: instalar rejas, mejorar iluminación, cambiar cerradura, contratar alarma, involucrar vecinos, etc., según aplique). Para los factores positivos, se refuerza la idea de que están ayudando (ej.: “las cámaras visibles que tienes disuaden a más de la mitad de los ladrones oportunistas, ¡sigue así! ⁷”). Este lenguaje debe motivar a tomar acción sin generar pánico, enfatizando la *prevención proactiva*.
- **Botones de acción post-resultado:** Debajo de la explicación, se incluirán CTA (Call to Action) relevantes, pensando en futuras extensiones: por ejemplo, un botón “**Ver recomendaciones detalladas**” que podría llevar a recursos o servicios profesionales (p. ej. enlace para contactar a una empresa de seguridad o asesor certificado). Otro botón “**Reevaluar**” para volver a llenar el formulario con otros datos (o si el usuario quiere probar distintos escenarios). También podría haber un botón “**Compartir mi resultado**” para redes (opcional, según requerimientos).
- **Diseño visual consistente:** Se aplicará un diseño **moderno** con Tailwind: probablemente un fondo claro o neutro, panel/cards con sombra suave para el resultado, y colores consistentes con la marca *Punto Seguro*. El dial multicolor será el elemento más llamativo. Todo el diseño será *responsive*, adaptándose de forma *mobile-first*: en móviles, el dial/score ocupará la mayor parte superior de la pantalla, y las explicaciones se verán en scroll debajo; en escritorio, podrían mostrarse lado a lado. Se cuidará la **accesibilidad de color** (por ejemplo, usar distintos iconos o patrones además del color para distinguir niveles, pensando en usuarios con daltonismo).

Microinteracciones y accesibilidad

La experiencia de usuario se enriquecerá con microinteracciones y se garantizará que la herramienta cumpla con estándares de accesibilidad **WCAG 2.1 AA**:

- **Efectos de foco/hover:** Los controles del formulario y botones tendrán estilos al pasar el ratón o al enfocarlos (para navegadores por teclado). Ejemplo: resaltar el borde o cambiar suavemente de color un botón “Siguiente” cuando el usuario lo apunta, indicando interactividad. Esto brinda feedback inmediato de qué es clicable. Todas las interacciones (hover, focus) estarán implementadas también de forma accesible (p. ej., usando `:focus-visible` en CSS para mostrar outline claro para quien navega con teclado).
- **Transiciones suaves:** Se implementarán animaciones sutiles al cambiar de una sección de formulario a la siguiente y al mostrar resultados. Por ejemplo, al hacer click en “Siguiente” sección, la nueva sección podría *deslizarse* hacia adentro o hacer un fade-in, en lugar de aparecer bruscamente. Igualmente, la animación del dial de resultado moviéndose, barras de progreso

llenándose gradualmente, etc., aportan una sensación pulida. Estas animaciones deben ser cortas (300ms–500ms) para no ralentizar la usabilidad, y estar programadas con CSS (utilizando utilidades de Tailwind para transitions) o pequeñas libs si es necesario.

- **Indicador de carga:** Al enviar el formulario completo para calcular el resultado, se mostrará un breve estado de *loading*. Puede ser un *spinner* animado acompañado del mensaje “Calculando resultado...”. Dado que la lógica de cómputo estará en el frontend (muy rápida), este estado de carga sería casi instantáneo; aun así, incluirlo mejora la percepción de respuesta (evita que el usuario piense que el botón no funcionó). Si se prevé que en futuras extensiones el cálculo involucre llamadas a API externas (por datos de barrio, etc.), este indicador será más necesario. El spinner desaparecerá dando paso a la pantalla de resultado final.
- **Tooltips informativos:** Se incorporarán **tooltips** (pequeñas descripciones emergentes) en ciertos textos o íconos para aclarar dudas sin saturar la interfaz. Por ejemplo, junto a la pregunta “¿planta baja?” podríamos tener un ícono “i” que al pasar el cursor diga “Las plantas bajas suelen ser más accesibles para intrusos que plantas altas, por eso preguntamos esto”. Estos tooltips aparecen con un ligero efecto fade o desplazamiento y cumplen directrices ARIA (atributos `aria-label` o `role="tooltip"` apropiados para lectores de pantalla). De esta forma, usuarios curiosos pueden profundizar en el “por qué” de cada pregunta, aumentando la percepción de rigor y transparencia.
- **Accesibilidad (WCAG):** Todo el desarrollo seguirá prácticas accesibles:
 - Colores con suficiente **contraste** para texto y elementos (p. ej., texto mínimo contraste 4.5:1 sobre fondo).
 - **Texto alternativo** en imágenes e iconos significativos. Por ejemplo, el dial de riesgo tendrá un `aria-label` que describa el puntaje (“Nivel de riesgo alto, 72 sobre 100”).
 - Navegación por **teclado** completa: el usuario puede recorrer campos con Tab, activar botones con Enter/Espacio, etc., en un orden lógico. Se gestionará el foco adecuadamente al pasar de preguntas a resultados (posicionándolo al inicio de la pantalla de resultado, por ej. en el heading del resultado).
 - **Etiquetas y ayudas en formularios:** Cada pregunta utiliza elementos semánticos de formulario (`<label>` claramente asociado al `<input>` correspondiente), facilitando la comprensión para tecnologías asistivas. Instrucciones adicionales (ej. unidades de medida, formatos esperados) se dan en texto visible o mediante `aria-describedby`. Según WCAG 2.1, se debe proporcionar instrucciones o etiquetas claras para que el usuario no se pierda ¹⁴ – por ello nos aseguramos de que cada campo esté identificado con un lenguaje simple.
 - **Responsive:** además de que el diseño se adapte a distintos tamaños de pantalla, se considerará la adaptación para *modo oscuro* si es posible (usando utilidades de Tailwind para dark mode), y el contenido seguirá siendo legible con fuentes escalables (usar unidades relativas como `rem` para que el usuario pueda hacer zoom sin romper el layout).
 - Se realizarán pruebas de usabilidad básicas con herramientas automatizadas (como Lighthouse, axe) para verificar cumplimiento WCAG.

En resumen, la app será **usable para cualquier persona**, incluyendo aquellas con discapacidad visual o motriz, y a la vez ofrecerá detalles interactivos sutiles que la hagan sentir *viva* y de calidad profesional.

Tecnología, arquitectura y extensibilidad

- **Framework y stack:** Utilizar **React** como framework front-end principal, lo que nos permite construir la interfaz de forma declarativa y manejar el estado del formulario y cálculo fácilmente. Se usará **Tailwind CSS** para el estilo: aprovechar sus utilidades para diseño responsive, spacing, colores, tipografías, etc., garantiza consistencia y rapidez en maquetado. La estructura será la de una SPA ligera que podría incrustarse en cualquier página (por ejemplo, integrándola en la landing page del producto mediante un `<div id="evaluador-root"></div>` donde React renderiza la app).
- **Componentización:** Dividir la UI en componentes React reutilizables y bien organizados:
 - Componentes para **Preguntas:** por ejemplo `<QuestionGroup title="Entorno">...</QuestionGroup>` que dentro mapeará sub-componentes `<Question>` o campos específicos. Esto facilita futuras modificaciones (agregar o quitar preguntas o secciones sin afectar otras partes).
 - Componente **ScoreCalculator** (lógica): podría ser una función separada (hook o utility) que reciba las respuestas y retorne el score + factores destacados. Implementado en JS/TS para fácil mantenimiento.
 - Componente **RiskDial:** renderiza el dial o barra de riesgo dado un valor. Puede reutilizarse si en futuras extensiones queremos mostrar sub-riesgos (ej. riesgo de intrusión vs riesgo de detección).
 - Componente **ResultCard:** para cada factor explicado en los resultados.
- De esta manera, la interfaz es modular. Además, se pueden aislar estilos con Tailwind (usando clases utilitarias en JSX) y, si se necesita personalización mayor, añadir estilos CSS adicionales o usar componentes Tailwind pre-diseñados (como panels, tooltips via Headless UI, etc.).
- **Estado y lógica de negocio:** React manejará el estado del formulario (podemos usar `useState` o un manejador como Formik si conviene, aunque dado que son pocas preguntas, probablemente `useState` sencillo o `useReducer` basten). Al cambiar respuestas, podríamos recalcular preliminarmente el score en background si se quisiera mostrar un *preview*, pero principalmente el cálculo final se dispara onSubmit. No se requiere backend ya que el cálculo es totalmente en cliente (no se procesan datos sensibles, solo recomendaciones). **Nota de privacidad:** advertir al usuario que la evaluación es orientativa y que no se guardan sus datos, a menos que en el futuro se integre login para guardar evaluaciones.
- **Embebible en landing:** La aplicación debe poder insertarse fácilmente en la web de producto de la empresa. Para ello, se puede preparar un *bundle* (usando herramientas como Webpack/Vite) que exponga el componente principal `EvaluadorPuntoSeguro` para incrustar. Una opción es proporcionar un snippet de inserción tipo:

```
<script src="evaluador.bundle.js"></script><div id="evaluador"></div><script>ReactDOM.render(<EvaluadorPuntoSeguro />, document.getElementById('evaluador'));</script>
```

. Alternativamente, convertirlo en un **Web Component** custom (con Stencil o similar) para aún más fácil integración – aunque inicialmente con React ya es suficiente. La app será ligera y cargará rápido en la landing (se puede hacer code splitting para que el grueso de la lógica cargue sólo cuando el usuario inicia el cuestionario, optimizando la performance de la página principal).

- **Preparado para futuras extensiones:** La arquitectura debe permitir **escalabilidad funcional**. Algunas extensiones previstas:

- *Importar datos externos:* por ejemplo, integrar una API de delincuencia o índice de criminalidad del barrio según el código postal. Esto podría automáticamente ajustar el puntaje o dar contexto adicional (“Tu barrio tiene un índice de robos un 20% mayor que la media nacional ¹⁵, por lo que recomendamos precaución extra”). La implementación actual dejará espacios para esto, por ejemplo, estructurando el cálculo en varias capas (factores propios de la vivienda + factor externo de zona). Usando React, podríamos incorporar un efecto `useEffect` que al ingresar la dirección o postal, consulte a una API (no parte del MVP pero planificado).
- *Servicios profesionales vinculados:* la app podría enviar los resultados a un portal donde un experto brinde asesoría. Para ello, desde ya podemos estructurar los datos de resultado en un objeto JSON serializable (score y detalle de respuestas) de modo que, con consentimiento del usuario, se pueda enviar via HTTP a un backend o mail. Igualmente, se puede habilitar que el botón “Contactar a un profesional” abra un formulario pre-llenado con los datos del riesgo, facilitando la comunicación.
- *Mantenimiento de la matriz de riesgo:* dado que las amenazas evolucionan (y las soluciones también), es posible que en un futuro se quieran ajustar las ponderaciones (por ejemplo, nuevos tipos de ataques o nuevos dispositivos de seguridad). Esto se simplifica centralizando la definición de factores en una estructura (como un JSON o una sección de código clara). Codex debe generar ese bloque de configuración claramente para fácil edición.
- *Multilinguaje:* inicialmente todo el texto estará en **castellano** (como solicitado), pero podríamos facilitar la traducción en el futuro separando los textos en un objeto de constantes o usando un framework i18n. De esta forma, extender a inglés/u otros sería cuestión de añadir archivos de traducción, aumentando el alcance de la herramienta.

En conclusión, este comando especifica detalladamente la creación de la herramienta *Evaluador Punto Seguro*. Codex debe utilizar estos lineamientos para generar una aplicación React/Tailwind completa: un formulario dividido en secciones con ~15 preguntas, interacciones amigables, cálculos basados en estándares de seguridad (CPTED, ISO 22341, datos de criminología) y una pantalla de resultados impactante y útil. Todo el contenido debe presentarse en español claro y profesional, transmitiendo confianza al usuario con fundamentos reales ¹⁶ ¹⁷. El resultado será una herramienta web embebible que ayude a los usuarios a conocer su nivel de riesgo de robo y a tomar medidas informadas para mejorar su seguridad. ¡Manos a la obra con la implementación!

1 ISO 22341:2021 - Designing Buildings

<https://www.designingbuildings.co.uk/wiki/ISO%2022341:2021>

2 16 Criminología ambiental: revolucionando la seguridad de tu vivienda

<https://genomadelrobo.com/blog-pildoras-de-seguridad/criminologia-ambiental-ciencia-revoluciona-seguridad-vivienda/>

3 5 6 9 Preguntas sobre cómo evitar el robo en viviendas - Soluciones de Seguridad Premium para Viviendas de Alto Standing | Ángel Olleros

<https://www.angelolleros.com/preguntas-como-evitar-robo-viviendas/>

4 ¿Cómo evaluar el riesgo de intrusión en las viviendas?

<https://es.linkedin.com/pulse/c%C3%B3mo-evaluar-el-riesgo-de-intrusi%C3%B3n-en-las-viviendas-genomadelrobo-jxkze>

7 11 Do Home Security Cameras Deter or Prevent Crime? | Security.org

<https://www.security.org/security-cameras/deter-crime/>

8 17 Rutgers Study Finds Alarm Systems Are Valuable Crime Fighting Tool | Rutgers University

<https://www.rutgers.edu/news/rutgers-study-finds-alarm-systems-are-valuable-crime-fighting-tool>

10 13 ¿Merece la pena instalar rejas en las ventanas? - MAPFRE

<https://www.mapfre.es/particulares/seguros-de-hogar/articulos/ventajas-rejas-ventanas/>

12 Niveles de seguridad en viviendas, clasificación según riesgo real.

<https://genomadelrobo.com/blog-pildoras-de-seguridad/niveles-seguridad-viviendas-clasificacion-segun-riesgo/>

14 Understanding WCAG 3.3.2 Labels or Instructions

<https://www.wcag.com/developers/3-3-2-labels-instructions/>

15 ¿Qué probabilidades tengo de sufrir un robo en mi casa? - Arrenta

<https://www.arrenta.es/blog/seguros-hogar-rc/probabilidades-sufrir-robo-casa/>