

PORTAL PARA P.I.A

PRESERVE
BLUR FACES

Reemplazo para Amazon Rekognition

José Manuel
Alonso Tirado

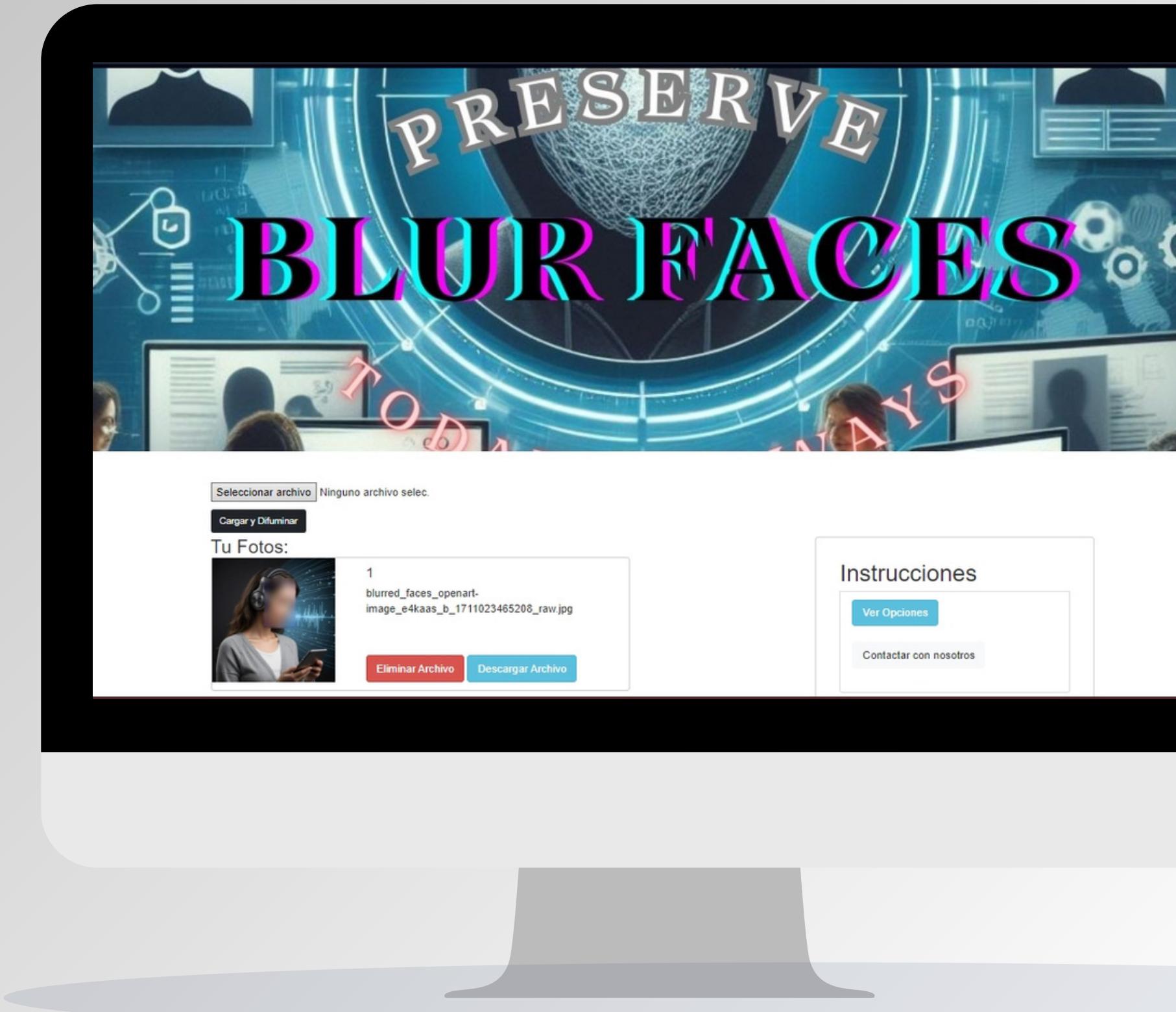


La Web: Blur Faces

Acabas de ver cómo utilizar el servicio Amazon Rekognition para detectar caras en imágenes. Ahora trabajas para un empresario increíblemente tacaño que prefiere gastar el dinero en otros menesteres.

Te ha encomendado rehacer la aplicación anterior para que no haga uso de los servicios de Amazon y solo use librerías o modelos open source.

¡Buena suerte!



Nota: he enfocado el ejercicio como una aplicación que pueda difuminar rostros para proteger la identidad en redes, por ejemplo, u otros usos.

Los puntos clave

1

código
python

app_bf.py

2

Portal B&F

Conexión con la página
para mandar el texto
y nos retorne audio

3

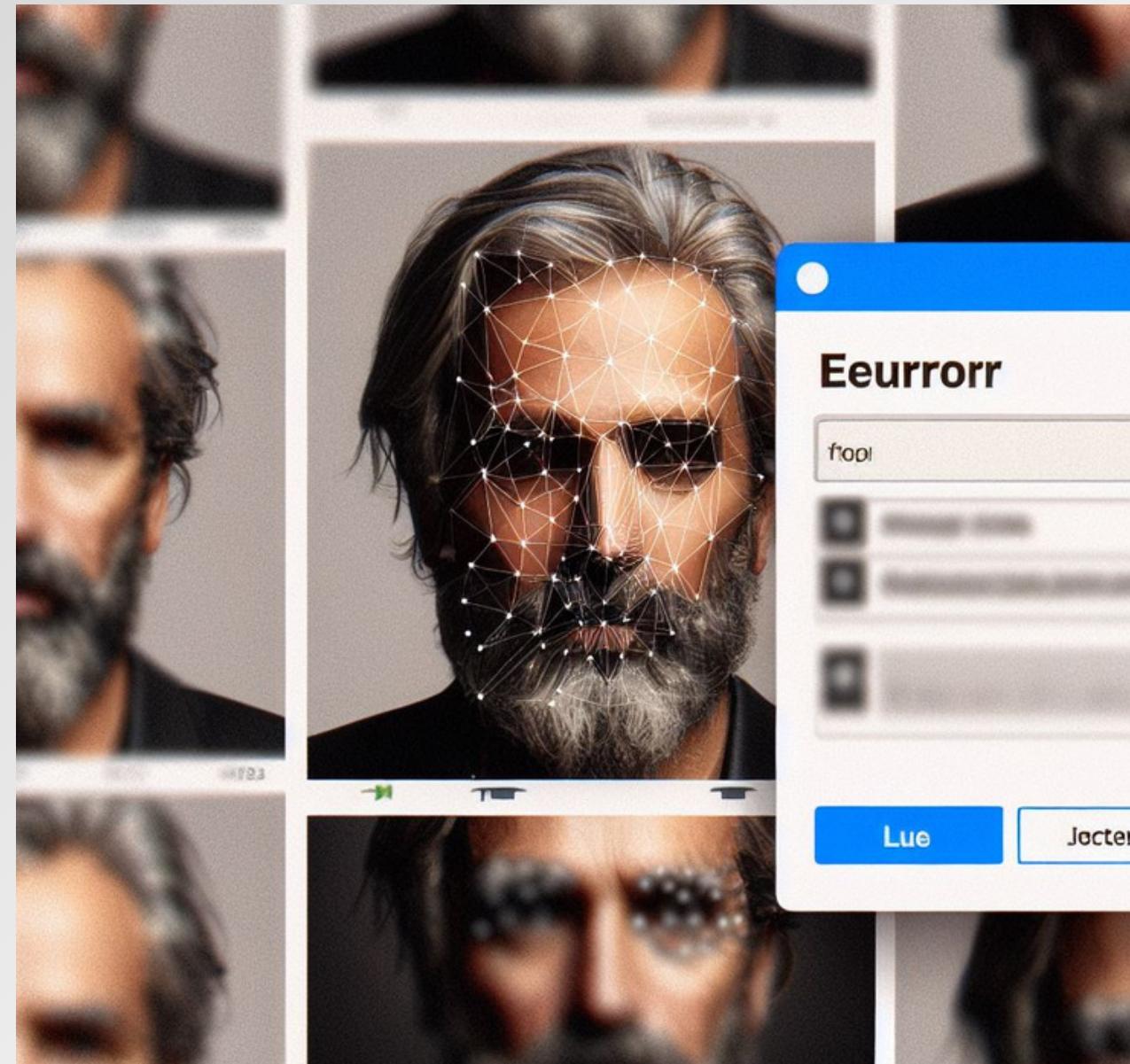
Sin Amazon Polly

He usado el Algoritmo MTCNN

Multi-task Cascaded
Convolutional Networks

Sin Usar el servicio Amazon Rekognition, para detectar caras en imágenes, nos decantamos por Multi-Task cascaded convolutional networks, la cual es capaz de detectar uno o múltiples rostros en imágenes o secuencias de video, entre otras opciones como:

- OpenCV: La biblioteca de visión por computadora de código abierto más utilizada, incluye métodos para la detección facial, como cv2.face
- MobileNet, también una Red Convolución capaz detectar objetos, pero que puede ser entrenada para la detección de rostros. La gran ventaja de esta red es su rapidez para el procesamiento, que se logra haciendo las convoluciones de una manera más eficiente(quizás para el próximo jefe no tan tacaño;)



MTCNN

Multi-task Cascaded Convolutional Networks

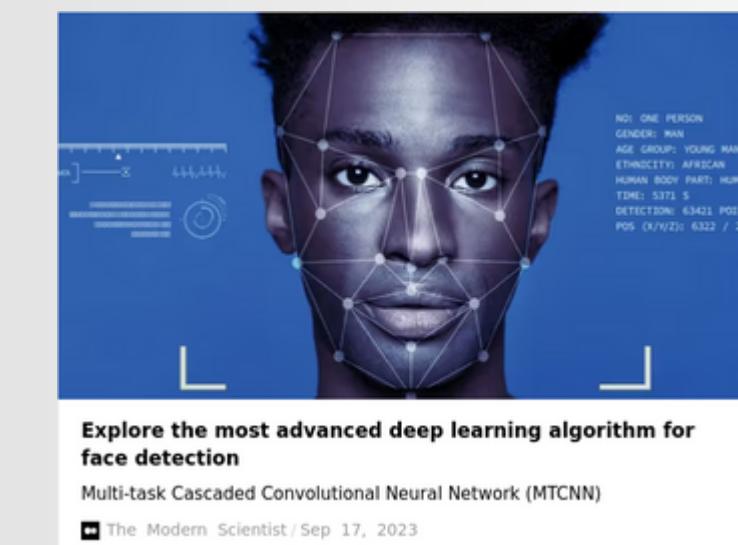
MTCNN es un algoritmo de detección facial que utiliza una red neuronal convolucional en cascada para realizar múltiples tareas relacionadas con la detección facial. Fue propuesto por Zhang et al. en 2016 y se ha convertido en una de las técnicas más populares y efectivas para detectar caras en imágenes.

El algoritmo MTCNN consta de tres etapas:

- Detección de rostros: Utiliza una serie de convoluciones para detectar candidatos a regiones faciales en la imagen.
- Refinamiento de rostros: Refina las regiones candidatas para ajustarlas mejor a las caras reales.
- Extracción de puntos clave: Identifica los puntos clave importantes en la cara, como los ojos, la nariz y la boca.

MTCNN es conocido por su precisión y eficiencia en la detección facial, incluso en imágenes con múltiples caras y en diferentes condiciones de iluminación y orientación.

En aplicaciones prácticas, MTCNN se utiliza comúnmente en sistemas de reconocimiento facial, análisis de emociones, seguimiento de objetos y otras aplicaciones donde la detección precisa de caras es fundamental.



MTCNN y mi proceso de ajuste

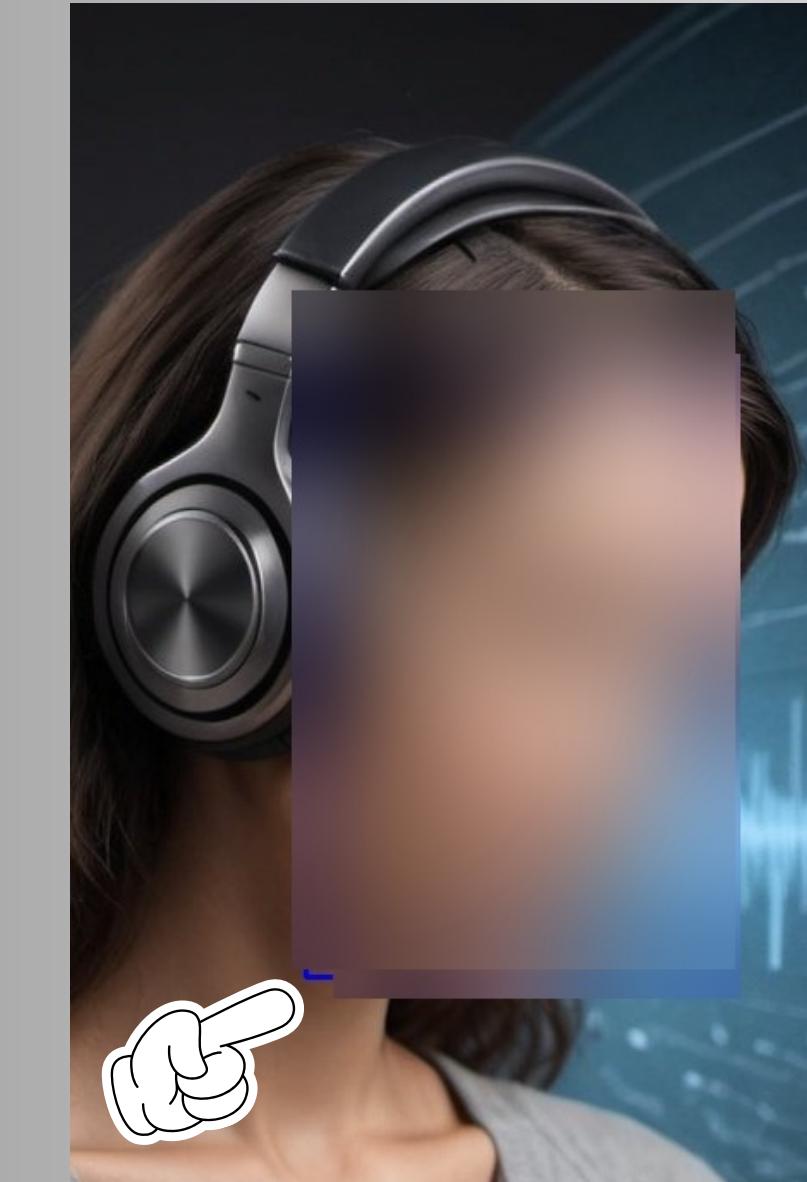


MTCNN y el resultado de dibujar el recuadro alrededor de la cara

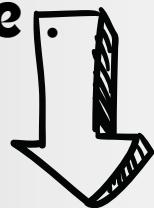


Al difuminar el rostro no cubría el recuadro y se veía en el resultado final

```
# Difuminar el área expandida del rostro
face = image[y:y+h, x:x+w]
blurred_face = cv2.GaussianBlur(face, (99, 99), 30)
image[y:y+h, x:x+w] = blurred_face
```



Detalle del ajuste previo del difuminado, se observa como queda un trozo azul aun sin cubrir, hasta ajustarlo correctamente.



```
# Definir un factor de expansión (por ejemplo, 0.2 para aumentar en un 20%)
expansion_factor = 0.001
# Calcular el aumento en las coordenadas
dx = int(w * expansion_factor / 2)
dy = int(h * expansion_factor / 2)
# Aplicar el aumento a las coordenadas del cuadro delimitador
x -= dx
y -= dy
w += 2 * dx
h += 2 * dy
# Asegurarse de que las coordenadas no sean negativas
x = max(0, x)
y = max(0, y)
```



Acceder a La Web

```
PS C:\pROGRM\0_0_BD_E_IA\CURSO_ESPEC_BD_E_IA\P_I_A\UNIDAD 2 Modelado IA\Aplicación web de predicciones (zalando edition)
hon.exe "c:/pROGRM/0_0_BD_E_IA/CURSO_ESPEC_BD_E_IA/P_I_A/UNIDAD 2 Modelado IA/Aplicación web de predicciones (zalando ed
2024-01-23 11:17:29.579277: I tensorflow/core/util/port.cc:113] oneDNN custom operations are on. You may see slightly di
ferent computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
WARNING:tensorflow:From C:\Users\ALBA_PTL\AppData\Local\Programs\Python\Python310\lib\site-packages\keras\src\losses.py:
se use tf.compat.v1.losses.sparse_softmax_cross_entropy instead.

2024-01-23 11:17:41.643709: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to
To enable the following instructions: SSE SSE2 SSE3 SSE4.1 SSE4.2 AVX2 FMA, in other operations, rebuild TensorFlow with
WARNING:tensorflow:From C:\Users\ALBA_PTL\AppData\Local\Programs\Python\Python310\lib\site-packages\keras\src\backend.py:
ase use tf.compat.v1.executing_eagerly_outside_functions instead.

WARNING:tensorflow:From C:\Users\ALBA_PTL\AppData\Local\Programs\Python\Python310\lib\site-packages\keras\src\optimizers
f.compat.v1.train.Optimizer instead.

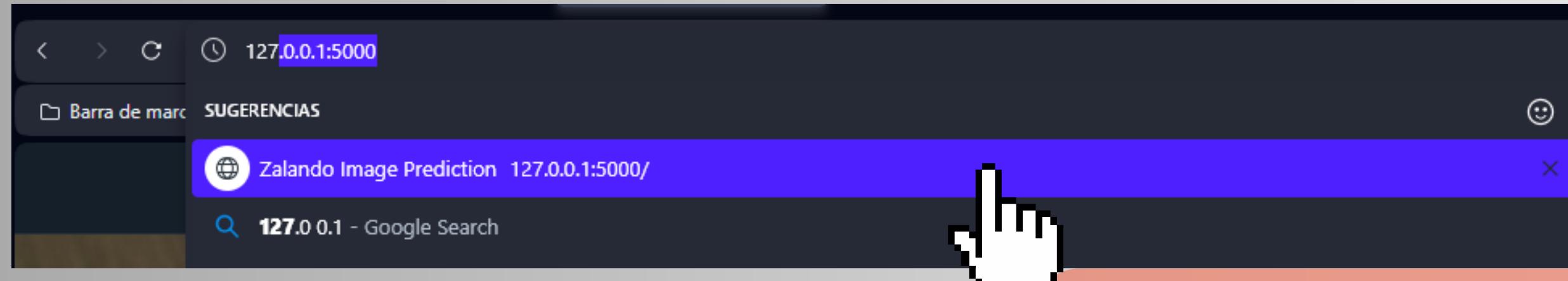
* Serving Flask app 'apw_zalando'
* Debug mode: True
WARNING: This [Seguir vínculo (ctrl + clic)]. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

Hacer Crtl+Click en 127.0.0.0:500

Al ejecutar el código Python te saldrá:

*Serving Flask “app_t2t.py”

La aplicación ya estará en marcha. Podremos acceder de dos formas diferentes, explicadas en las imágenes.

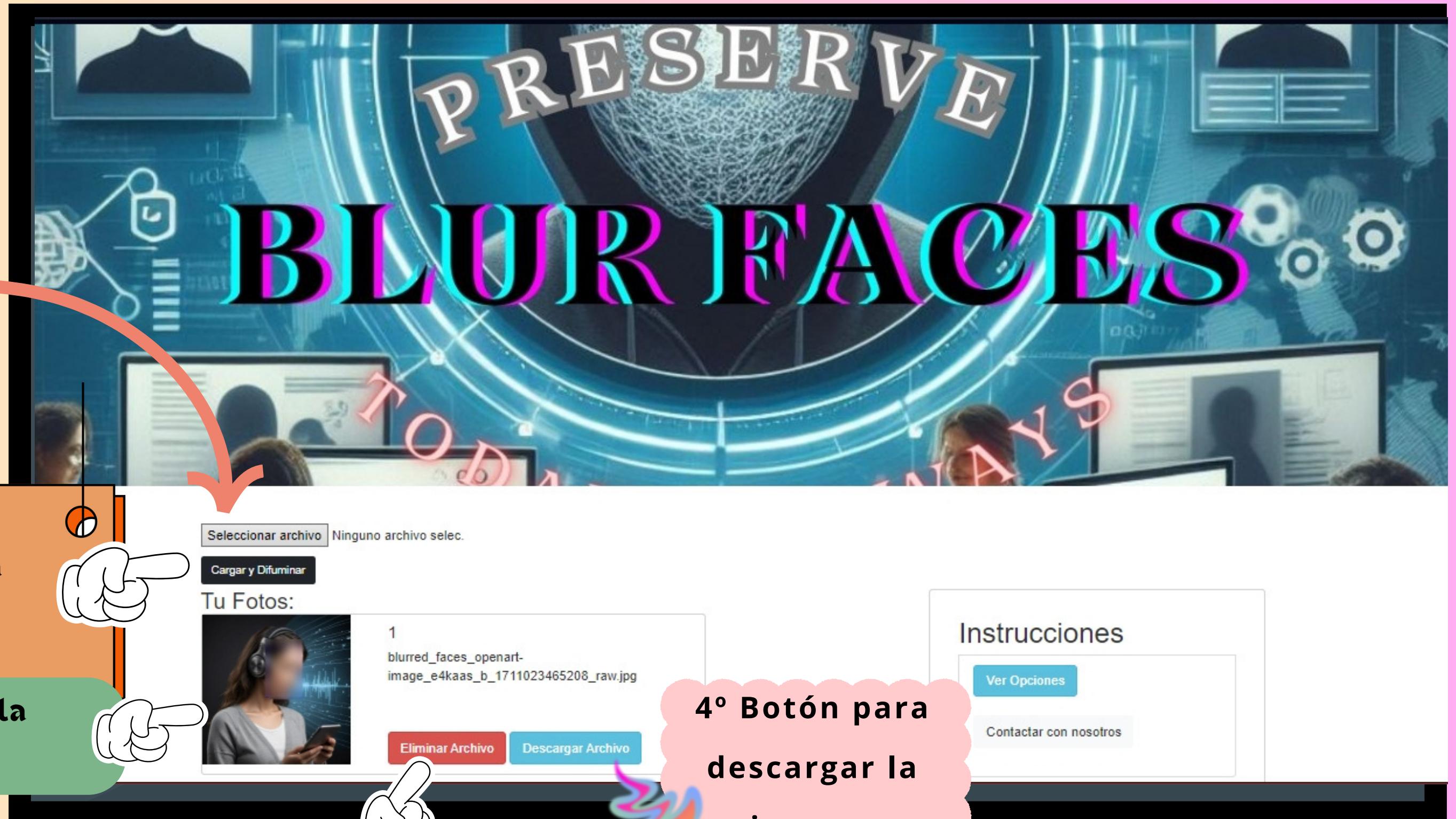


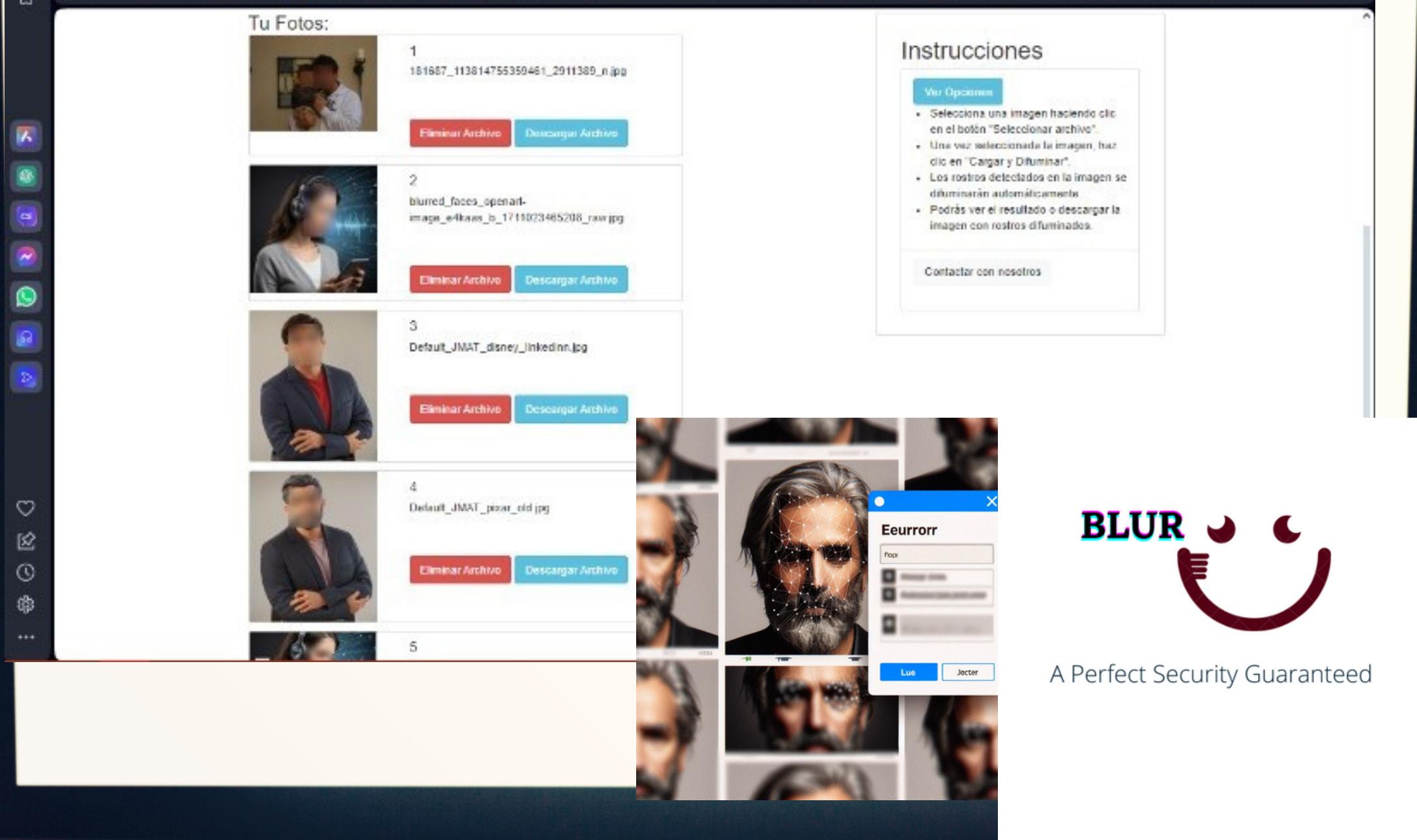
1º Para Cargar la imagen que queremos procesar

2º Botón para enviar la imagen para procesar

3º El resultado de la imagen procesada.

5º Botón para borrar la imagen, si no es de tu gusto.





La página, tiene todo el proceso de maquetación usando Inteligencia Artificial, como la imagen de Portada, el logo que a la vez hace de favicon, la pagina de ERROR.



**Imágenes generadas por IA al introducir el
pront página web para detección y
difuminado de imágenes que la convertí en la
página de error cuando no es correcto el
archivo**

Gracias

JMAT@HOTMAIL.ES