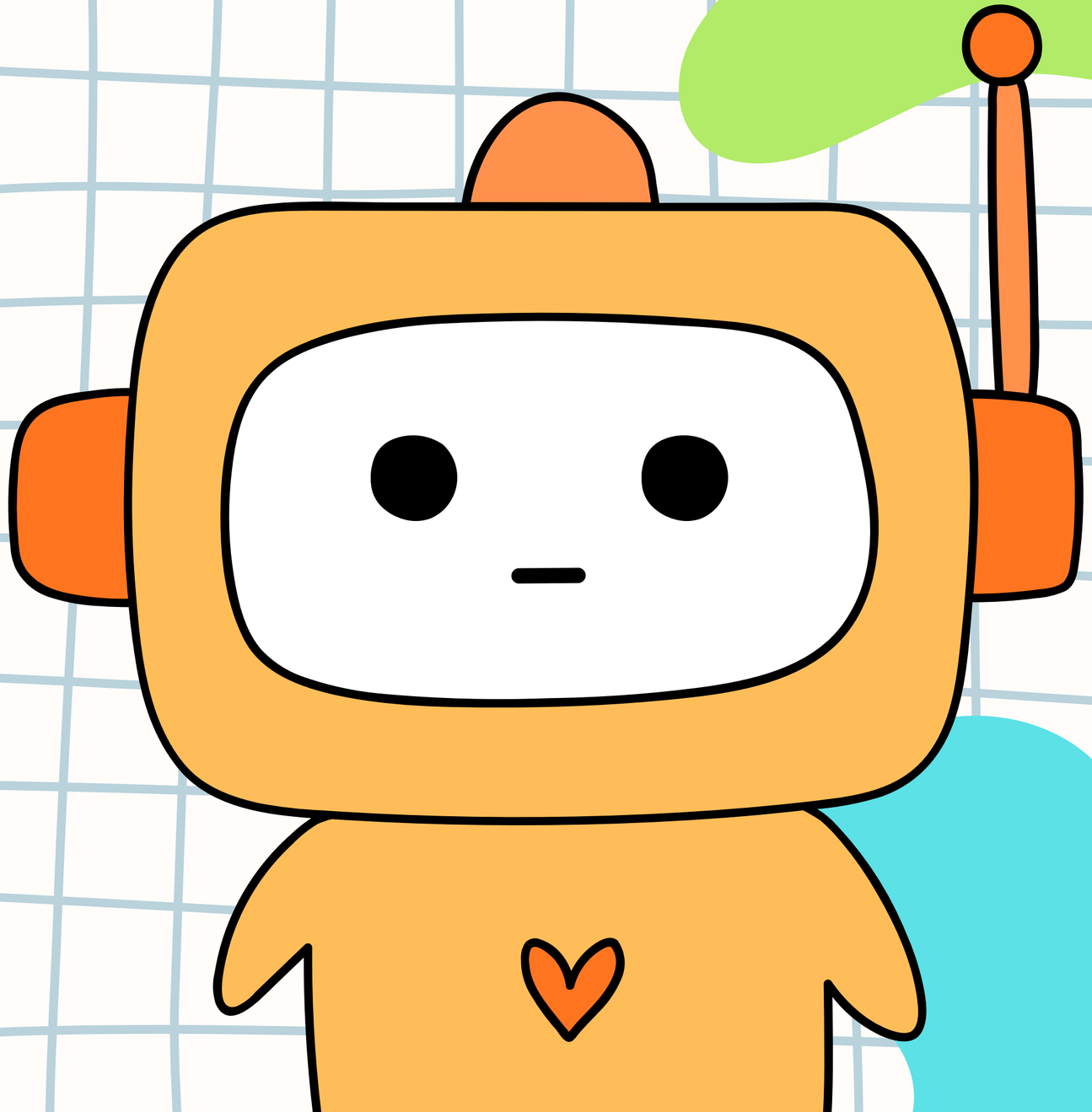
Lector QR

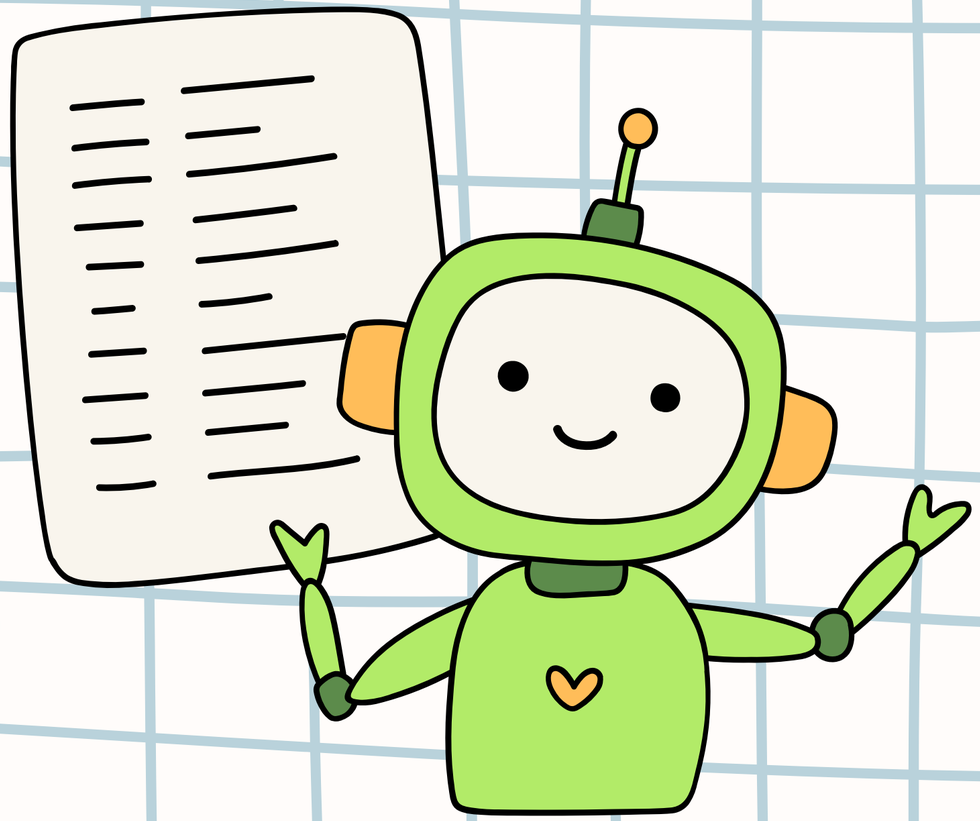


PROBLEMATICA

Poder acceder de manera rapida a los qr dentro de un inventario, para poder tener un mejor control de los datos

RESPUESTA

- Procesar imagenes con QR guardados en el banco de imagenes.
- Lectura de códigos QR en tiempo real.
- Filtros para mejorar el contraste y eliminar el ruido.
- OpenCV para detectar y decodificar el QR.



LIBRERIAS UTILIZADAS

```
#Importamos librerias
import cv2 as cv
import numpy as np
import os
```

PROCESAMIENTO DE IMAGEN
&
DETECCION QR

OPERACION DE MATRICES
&
KERNEL

VALIDACION DE EXISTENCIA DE ARCHIVOS

FILTROS

```
gray=cv.cvtColor(img1,cv.COLOR_BGR2GRAY)
```

CONVERTIR A ESCALA DE GRISES

```
_, mask = cv.threshold(gray, 100, 255, cv.THRESH_BINARY)
```

BINARIZACION DE LA IMAGEN

```
mask_blur = cv.GaussianBlur(mask, (3,3), 0)
```

REDUCIR RUIDO

```
#aplicar erosion y dilatación para resaltar bien las lineas del qr
kernel = np.ones((5,5), np.uint8)
mask_erode = cv.erode(mask_blur, kernel, iterations=1)   # Erosión
mask_dilate = cv.dilate(mask_erode, kernel, iterations=1) # Dilatación
```

RESALTAR CODIGO

ARCHIVOS ENTRADA

- Imagen en .jpg
- Imagen en tiempo real
- Texto para escoger que imagen guardada.

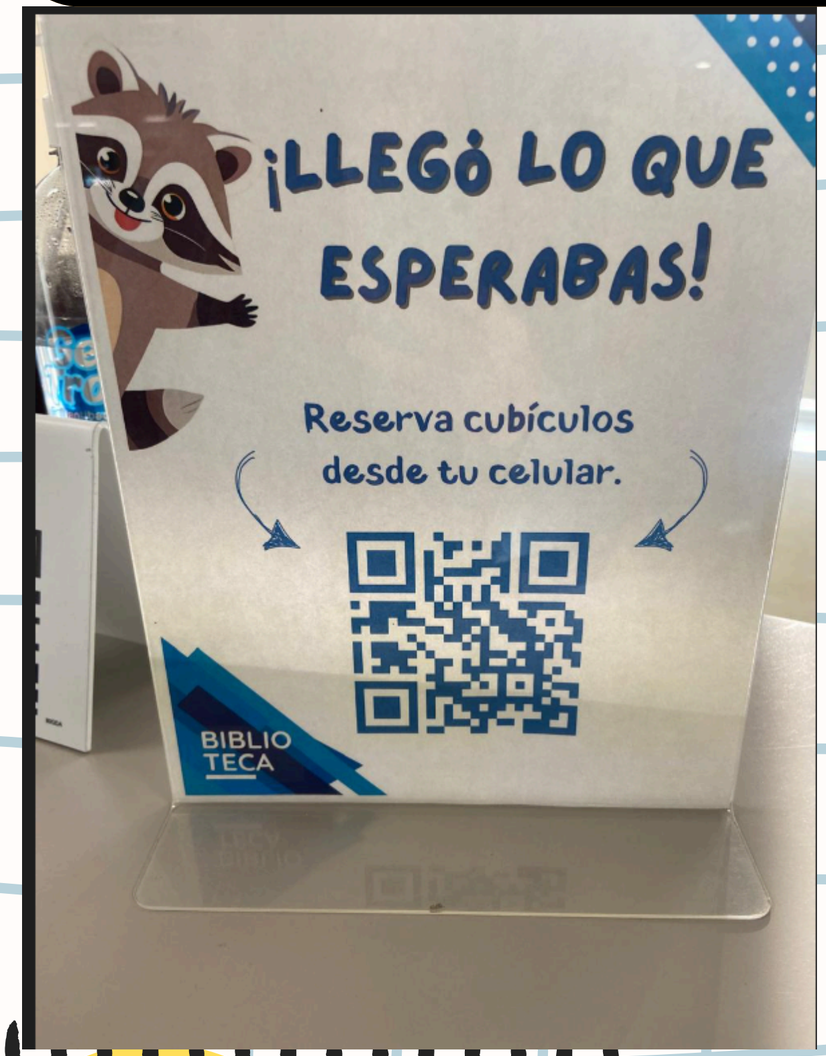
ARCHIVOS SALIDA

- Imagen procesada
- Contenido decodificado del QR

TERMINAL

```
Introduce 1 para leer el QR de una imágen o 2 para leer los que estén en la cámara:
1
Introduce el nombre de la imagen que quieras procesar:
qr
Imagen guardada :D
Código QR detectado, información guardada en la imágen original
Introduce 1 para leer el QR de una imágen o 2 para leer los que estén en la cámara:
```

ARCHIVOS ENTRADA



ARCHIVOS SALIDA

CONCLUSION

Realizar este script nos fue útil para entender los elementos que integran un QR y con ello saber como hacer que nuestro programa los interpretará correctamente y así el usuario


THE END